

絵でわかる

たのしい

パソコン入門

MSX

関口泰夫



新星出版社

新星出版社の
パソコン・マイコンシリーズ

MSXシリーズ

定価各980円

めはてじの
パソコン入門

山下 利秋著

めはてじの
プログラミング

大沢昭二／田中一郎著

でゲーム
覚える
MSXベーシック

川中 篤胤著

よく
わかる
ぼくらのパソコン入門

山下 利秋著

MSXベーシック用語辞典

田中一郎／小山郁夫著



**絵でわかる
たのしいパソコン入門**

新星出版社

●はじめにっておきたいこと

MSXタイプのコンピュータの^{にんき}人気は、やっぱりすごいもののようです。^{しょうがくせい}小学生の^{ひと}人もおとなの^{ひと}人も、いちどMSXに^て手を^ふ触れてしまえば、すっかりとりこになってしまうのでしょう。

ほかのタイプのパソコンとくらべてみても、^{きのう}機能もおとるところはないようですし、それでいて^ね値^{やす}だんは安いというのですから、^{にんき}人気のほどもうなずけますね。それに、なんといっても、^{ベーシック}MSXBASICという^{きょうつう}共通のことばを^も持っているのがいいのです。MSX用のソフトウェアなら、^{じぶん}でも自分のマシンで^{つか}使えるのですから、パソコン^{にゅうもんよう}入門用のマシンとして、いちばん^{てき}適しているといってもいいのではないのでしょうか。

MSXは、それほどすばらしいマシンなのです。これを^{あそ}遊ばせておく^て手はないのです。うんと^{つか}使いまくってやりましょう。

ただ、パソコンにはいろいろ^{やくそく}約束ごとがあったり、^{ベーシック}BASICという^{えいご}英語みたいなものがあったりして、めんどくさいなあ、なんて^{おも}思わないでください。また、^{ベーシック}BASICをマスターしてからキーをたたこうなんて、^{けっ}決して^{おも}思わないでください。

はじめは、なにもわからなくてもいいのです。いわれたとおりにキーをたたくことです。よくわからないページがあったら、とばして^{さき}先に^{すす}進むのもいいでしょう。おぼえようとしても、なかなかおぼえられないのがコンピュータですし、おぼえようとしなくとも、キーをたたいているうちに、^{しぜん}自然にわかってしまうのもコンピュータなのです。とにかく、やってみることをおすすめします。

(著者しるす)

も く じ



パート 0 コンピュータってな～に？

- 0-1 こんにちはパソコン** 10
パソコンの^{つか}使われ方^{かた}いろいろ(11)
- 0-2 家^{いえ}の中^{なか}にはコンピュータだらけ** 12
身^みのまわりにあるコンピュータ(13)
- 0-3 楽しいディスプレイ^{たの}** 14
パソコン^が画面^{めん}表示^{ひょうじ}のいろいろ(15)
- 0-4 パソコンってどんなマシンなの？** 16
パソコンの^{こうせいひん}構成品(17)
- 0-5 パソコンのしごと** 18
パソコンのシステム(19)
- 0-6 パソコンは0か1だけの世界^{せ かい}** 20
10^{しんすう}進数と2^{しんすう}進数(21)
- 0-7 ビットとバイト** 22
各^{かく}ビットの^{じょうほうりょう}情報量(23)
- 0-8 ぼくのパソコンだ たいせつに扱^{あつか}ってね** 24
パソコンの^{たいてき}大敵^{だいてき}は？(25)
- 0-9 正^{ただ}しく接^{せつ}続^{ぞく}しよう** 26
周辺^{しゅうへん}機器^きの^{せつぞく}接続^{せつぞく}(27)
- 0-10 いよいよスイッチONだ** 28
カーソル^{つか}キー^{かた}の^{つか}使い方^{かた}(31)
- 0-11 キーボードから文字^{も じ}や記号^{き ごう}の入力^{にゅうりょく}** 32
文字^{も じ}や記号^{き ごう}の入力^{にゅうりょく}方法^{ほうほう}(33)／ファンクション^{つか}キー^{かた}の^{つか}使い方^{かた}(34)
- 0-12 プログラムは保存^{ほ ぞん}しておこう** 35
カセット^{カセット}セーブ^{セーブ} カセット^{カセット}ロード^{ロード} ●保存^{ほ ぞん}したり読み^よ込^こんだり(35)／カセット^{カセット}セーブ^{セーブ}めい^{めい}れい^{れい}命令^{めいれい}とCLOAD^{カセットロードめいれい}命令^{めいれい}の^{つか}使い方^{かた}(36)

パート 1 パソコン操作への第一歩

- 1-1 パソコン^{けいさん}計算機** 38
 PRINT命令^{プリント}その1 ● 計算^{けいさん}して答え^{こた}を画面^{がめん}に表示^{ひょうじ}する(38) / 計算問題^{けいさんもんだい}
 のいろいろ(39) / CLS命令^{クリアスクリーン} ● 画面^{がめん}をきれいにそうじしてから(40) /
 プログラム^{つく}を作ってみよう(41)
- 1-2 画面^{がめん}に文字^{もじ}を描く** 42
 PRINT命令^{プリント}その2 ● 文字^{もじ}やキャラクタ^かを画面^{がめん}に表示^{ひょうじ}する(42) / アル
 ファベットや模様^{もよう}を書こう(43) / LIST命令^{リスト} ● プログラム^{がめん}を画面^{がめん}に
 表示^{ひょうじ}する(44) / 打ちまちがえた字^うを正しい字^{ただ}に直す^な(45) / 文字^{もじ}や記
 号^{ごう}を追加^{ついか}する(46) / まちがえた文字^{もじ}を消す^け(47)
- 1-3 好きな位置^いに表示^{ひょうじ}する** 48
 LOCATE命令^{ローケート} ● 画面^{がめん}の縦^{たて}の位置^いと横^{よこ}の位置^いを指定^しする(48) / ひと
 つのマス目^めに1字^じ表示^{ひょうじ}(49) / グラフィックキャラクタ^{ひょうじ}を表示^{ひょうじ}するプ
 ログラム(50)
- 1-4 データ^いを入れる箱^{はこ}** 51
 = (イコール^{きごう}記号)は「等しい^{ひと}」ではなく「代入^{だいにゅう}する」の意味^い(51) / 変
 数^{すう}は変化^{へんか}する数^{かず}、Aの値^{あたい}はどんどん変わる(53) / 文字変数^{もじへんすう}は変数名^{へんすうめい}
 のうしろに\$ (ドルマーク)をつける(54) / 文字変数^{もじへんすう}に代入^{だいにゅう}された
 数字^{すうじ}は数^{かず}ではなく文字列^{もじれつ}だ(55)
- 1-5 クルマ^{はし}を走らせる** 56
 LOCATE、PRINT、変数^{へんすう}のまとめプログラム(56) / クルマ^{がめん}が画面^{がめん}
 の上^うを走る^{はし}、走る^{はし}、走る^{はし}(58)
- 1-6 プログラム^{にゅうりよく}を入力する前に** 59
 NEW命令^{ニュー} ● メモリ^{なか}の中のプログラム^けを消してスッキリメモリ(59)
 AUTO命令^{オート} ● 行番号^{ぎょうばんごう}を自動的^{じどうてき}につけてくれる(59) / WIDTH命令^{ウイース} ●
 画面^{がめん}に表示^{ひょうじ}させる桁数^{けたすう}を決める(60)

パート 2 BASICをマスターしよう

- 2-1 とびとびにプログラム^{じっこう}を実行** 62
 GOTO命令^{ゴーツ} ● 指定^しされた行^{ぎょう}にジャンプする(62) / GOTO文^{ゴーツ}ショート
 プログラム(63)

- 2-2 ジャンプしてふたたび戻ってくる** 64
ゴ ー サ ブ めいれい リ タ ー ン もど つぎ めいれい
 GOSUB命令●RETURNで戻って次の命令から(64)／GOSUB文シ
 ョートプログラム(66)
- 2-3 あきずに同じ動作をくり返せ** 68
フ オー ネ ク ス ト おな どう さ かえ
 FOR～NEXT●くり返す動作の回数を指定して(68)／FOR～NEXT
 文ショートプログラム(69)／FOR～NEXTの中にまたFOR～NEXT
 (70)／FOR～NEXTのループ回数に注意する(71)
- 2-4 天気だったら野球 雨だったら勉強だ** 72
てん き や きゅう あめ べんきょう
 IF～THEN●F～THEN～ELSE(72)／IF THEN文の使い方(73)
 ／IF～THEN文ショートプログラム(74)
- 2-5 並んだデータを順番どおりに呼び出す** 77
なら リ ー ド デ ー タ じゅんばん よ だ
 READ～DATA●2つそろっていちにんまえ(77)／虹の色を塗る順
 番も決まっている(78)／READ～DATA文ショートプログラム(79)／
 READ～DATAのかたち(80)／電話管理プログラム(81)
- 2-6 まちがいの意味と行番号を表示** 83
い み ぎょうばんごう ひょうじ
 エラーメッセージ●プログラムを直せばいい(83)／よく出るエラー
 メッセージ(84)

パート3 やさしいプログラミング

- 3-1 プログラミングとは** 86
し ごと なが ず か
 フローチャート●仕事の流れを図に書いてみるとよくわかる(86)／
 フローチャートの書きかた(87)／INPUT命令●キーボードからデ
にゅうりよく
 ータを入力する(88)／三角形の面積を計算するフローチャート(89)
- 3-2 仕事の流れが違うとき** 91
し ごと なが ちが
すう すう くら べつ
 ぐう数かき数かを区別するフローチャート(92)
- 3-3 1から100までのたし算** 94
へんすう つか さん
 変数を使って1から100までのたし算をする(95)
- 3-4 サイコロころころ** 98
ランダム かんすう インデジャーかんすう
 RND関数とINT関数●2つそろわなければ整数は作れない(98)／0
いじょう みまん かず はっせい
 以上1未満の数を発生させてから(99)／小数点以下はとってしまう
 (100)
- 3-5 サイコロ応用ゲーム** 102
おうよう

ゲームプログラム ● サイコロコロコロ

3-6 **さんすうドリルプログラム** 105

さんすうドリルのフローチャート(106) / さんすうドリルのプログラム(107)

3-7 **プログラムの整理** 111

リ ナ ン バ ー め い れ い
RENUM 命令 ● プログラムの行番号を整理する(111) / DELETED
め い れ い
命令 ● プログラムを行番号ごととってしまう(111)

パート 4 図や絵を描いたり音を出したり

4-1 **グラフィックの画面作りでスタート** 114

ス ク リ ー ン め い れ い
SCREEN 命令 ● どんな画面にするかを定める(114) / グラフィッ
ク画面(115)

4-2 **図や絵に色をつける** 116

カ ラ ー め い れ い ぜん けい しょく はい けい しょく しゅう へん しょく き
COLOR 命令 ● 前景色、背景色、周辺色を決める(116) / 絵とき
カ ラ ー め い れ い
COLOR 命令(117) / COLOR 文ショートプログラム(118)

4-3 **直線で描くグラフィックス** 119

ラ イ ン め い れ い せん ひ はこ か はこ ぬ ラ イ ン ぶん
LINE 命令 ● 線を引き、箱を描き、箱を塗る(119) / LINE 文ショ
ートプログラム 1 (121) / LINE 文ショートプログラム 2 (123)
／LINE 文ショートプログラム 3 (124)

4-4 **円、だ円、扇形を描く** 125

サ ー ク ル め い れ い えん か
CIRCLE 命令 ● だ円も描けるパソコンのコンパス(125) / CIRCLE
文ショートプログラム 1 (127) / CIRCLE 文ショートプログラム
2 (129) / CIRCLE 文ショートプログラム 3 (131)

4-5 **線で囲んだ領域に色を塗る** 132

ペ イ ン ト め い れ い きょう かい しょく かこ ぼ しょ りょう いく しょく ぬ
PAINT 命令 ● 境界色で囲まれた場所を領域色で塗る(132) / PAINT
文ショートプログラム(135)

4-6 **ドットに色をつけて表示させる** 136

ポ イ ン ト セ ッ ト め い れ い し て い ざ ひょう てん ひょう じ ポ イ ン ト セ ッ ト
PSET 命令 ● 指定した座標の点を表示(136) / PSET ショートプロ
グラム(137)

4-7 **点を消す方法で点を表示させる方法** 138

ポ イ ン ト リ セ ッ ト め い れ い し て い い ち てん け ほう ほう
PRESET 命令 ● 指定した位置の点を消す(138) / PRESET ショー
トプログラム(139)

4-8	パソコンのペン描き <small>が</small> 140
	<small>ド ロ ウ め い れ い</small> DRAW 命令 ● 画面に自由な図形を描く (140) / <small>ド ロ ウ ぶん</small> DRAW文の文字式 (141) / <small>ド ロ ウ ぶん</small> DRAW文ショートプログラム (142)
4-9	グラフィック画面に文字を描く <small>が め ん</small> 143
	<small>オ ー プ ン め い れ い</small> OPEN 命令 ● ファイルを開く (143)
4-10	パソコン・ミュージック 144
	<small>プ レ イ め い れ い</small> PLAY 命令 ● “ ” で囲んだメロディーを演奏する (144) / C D E F G A B (145) / 音の長さ <small>おと</small> と休みの長さ <small>やす</small> (146) / オクターブ、テンポ、ボリューム (147) / 音の長さ <small>おと</small> 、オクターブは変わるたびに指定する (149) / トルコ行進曲 <small>こうしんきょく</small> のプログラム (151)
4-11	楽しいサウンド作り 153
	<small>サ ウ ンド め い れ い</small> SOUND 命令 ● サウンドを発生させる (153) / <small>はっせいげん</small> 発生源のレジスタ番 <small>ばん</small> 号とデータを指定して組み合わせる (154) / サウンド作りプログラム (155)

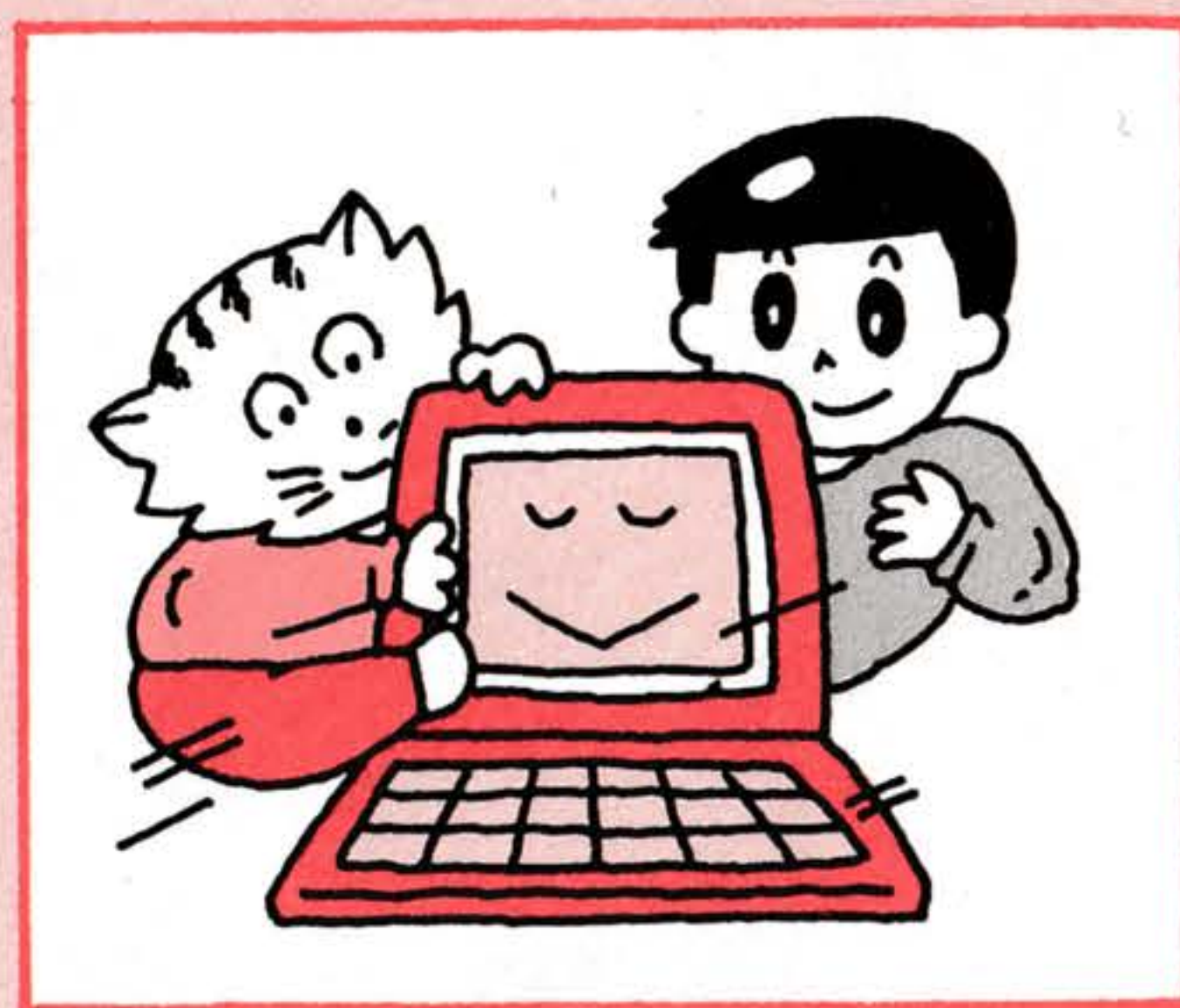
パート 5 ゲームで遊ぼう

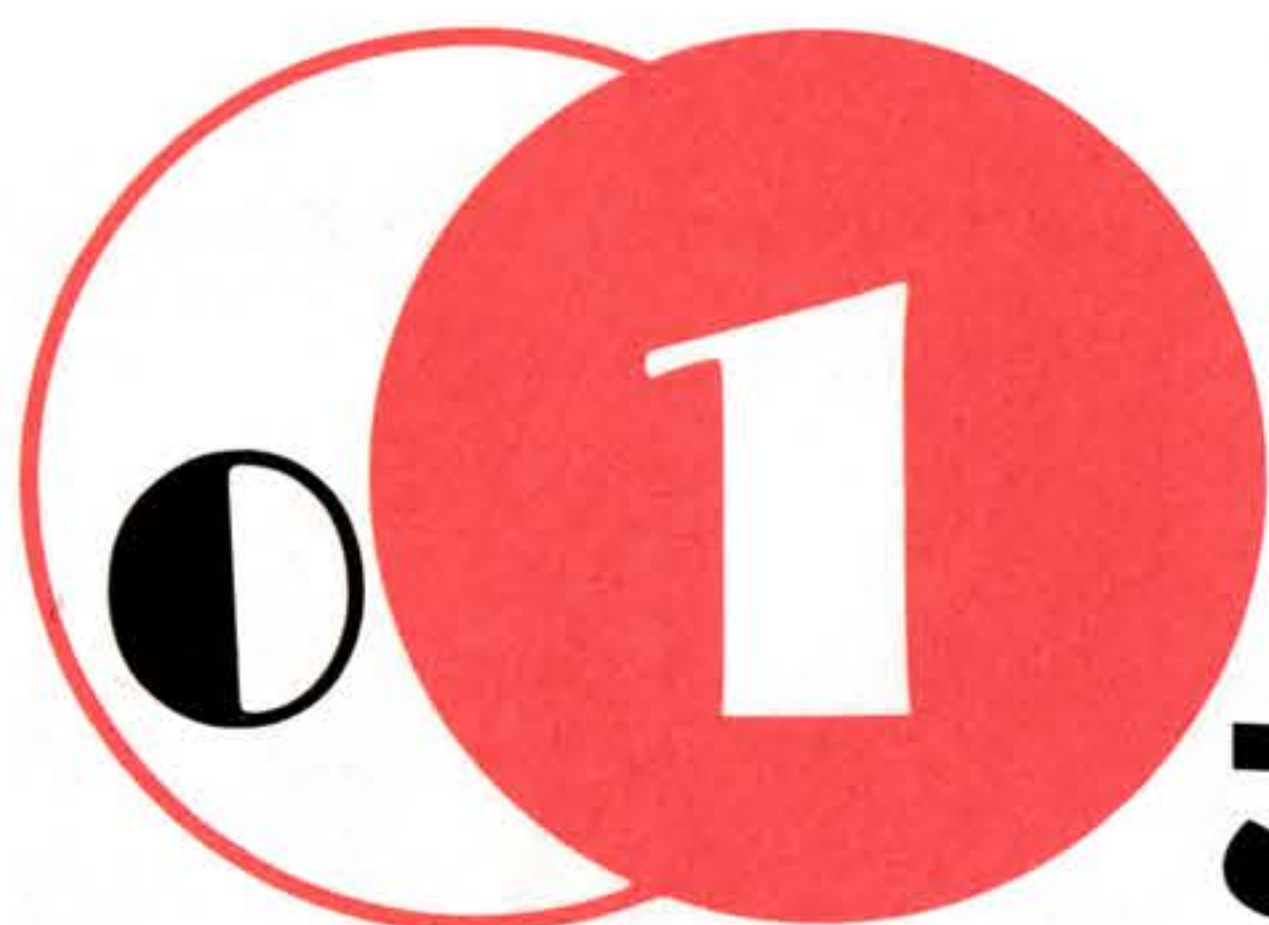
5-1	8×8のキャラクタを動かす <small>うご</small> 158
	スプライトパターン ● ゲームのコマ作り (158) / 8×8のスプライトパターンを定義する (159) / スプライトパターンを表示する (161) / <small>フ ッ ト</small> PUT SPRITE 命令 ● スプライトパターンを動かす (162)
5-2	16×16のインベーダーだ 165
	16×16のパターンを定義する (166) / インベーダーが左から右へ歩くプログラム (167)
5-3	3次元迷路脱出ゲーム 168
	ゲームプログラム ● 3次元迷路からの脱出 (168)
5-4	キーボード操作のトレーニング 176
	トレーニングのメニューは4つ (176) / トレーニングプログラム ● キーボードマスター (177)

☆ 索引	187
-------------	------------

パート0

コンピュータって な～に？





こんにちはパソコン



ゲーム機^きとしても楽しいけれど^{たの}

いまはもう、パソコンを^{ぜんぜん}知らない人^{ひと}なんて、ほとんどいないはずだ。でも、パソコンっていうことばから、なにを^{おも}う浮かべるかは、みんな^{ちが}違っているかもしれない。

いろ色あざやかなパソコンゲームやテレビゲームの画面^{がめん}を^{おも}う浮かべる人^{ひと}、むずかしいさんすうの^{けい}計算式^{さんしき}を^{れんそう}連想する人^{ひと}、文章^{ぶんしょう}を^{いんじ}どんどん印字していくワープロのようなものを^{そうぞう}想像する人^{ひと}と、いろいろあるはずだ。

ホームコンピュータ^{きゅう}級の^こ小型^{がた}パソコンでも、使^{つか}い方は^{かた}数^{かず}かぎりないといってい^いくらいあるのだ。

たの楽しく^{あそ}遊ぶ^{よう}ホビーの用途^とだけでも、ゲームのほか、ゲームソフトやレコード、ビデオテープなどのデータバンクや、アニメーション、音響^{おんきょう}、カラーグラフィックスなど、いくらでもあるのがパソコンの^せ世界^{かい}といってい^いだろう。

もちろんホビーだけじゃないよ。会社^{かいしゃ}の事務^{じむ}だってこなすし、勉強^{べんきょう}の手伝い^{てつだ}もOKという^{ばんのう}万能マシン^{かんが}なのだと考えておこうよ。

パソコンは
こっちでなにも
させなければ
なにもしないんだよ



ゲーム機^きだけが
パソコンじゃないさ
さんすうだって
図画^{ずが}だって
代わり^かに
やってくれるかもね

●パソコンの使われ方いろいろ

●家庭では

ゲームするだけが能^{のう}じゃない。
ダイエットメニューを作^{つく}ったり電^{でん}
話帳^{わ ちやう}を作^{つく}ったり。



●学校や塾では

教育用^{きやういくよう}の利用範囲^{り ようはん い}はどんどん大
きくなる。各課目^{かく か もく}の辞書^{じ しょ}としての
役割^{やくわり}や得点計算^{とくてんけいさん}、順位計算^{じゆん い けいさん}などお
手^てのものだ。

●オフィスでは

営業^{えいぎやう}や事務管理^{じ む かん り}だけでなく、端^{たん}
末^{まつ}としてもデータバンクとしても
オフィスオートメーションには欠^か
かせないマシンになっている。

パソコン応用分野の一部^{おうようぶん や いち ぶ}

(ホームコンピュータ)ゲームマシン／ホビー用^{よう}データバンク／オー
ディオ・ビデオ制御^{せいぎよ}／家計管理^{か けいかん り}／食事献立表^{しょく じ こん だてひやう}／家電制御^{か でん せいぎよ}／その他^た

(教育補助マシン)各課目^{かく か もく}ごとの辞書^{じ しょ}／視覚教育教材^{し かくきやういくきやうざい}／理数計算補
助機^{じよ き}／教育用シミュレータ^{きやういくよう}／教材データバンク^{きやうざい}／その他^た

(オフィスオートメーション)ワードプロセッサ／事務・営業管理^{じ む えいぎやうかん り}
／事務データバンク^{じ む}／一般事務機^{いっぱん じ む き}／端末機^{たんまつ き}／その他^た

②

家の中には コンピュータだらけ



め み
目には見えないけどはたらきつづけ

タイプライターみたいなキーボードと、テレビの画面がめんそっくりなディスプレイと、紙に文字を印刷さつするプリンタがそろっていれば、だれだって、「ははあ〜ん、これがコンピュータだな！」ってわかっちゃう。

だけど、ちょっと見たところ、コンピュータではないのに、ほんとうは内部ないぶにコンピュータが入っているものは、台所だいどころにもダイニングルームにも、勉強部屋べんきょう べやにも、たくさんころがっているんだよ。

たとえば、おかあさんがだいじにしているミシン。花柄はながらでもほかの模様もようでも、どんどんししゅうしてしまうのも、中なかに入っているコンピュータがやらせているのだ。

ほかにもいろいろある。かつてに走ったり曲まがったり止まったりするプラモの戦車せんしゃ。暑あつかったら空気を冷やし、寒さむくなったらあたためる部屋の中のエアーコンディショナー、洗せんたく機きなどなど。みんなコンピュータが命令めいれいして動かうごしているものが多いのだ。

へ や なか いち ど
部屋の中をもう一度
見わたしてみよう
コンピュータが
はい せいひん
入っている製品は
たくさんごろごろ
ころがっているよ



あたま なか
頭の中に
パソコンを入れれば
さんすうの宿題しゅくだいなんか
イチコロだ



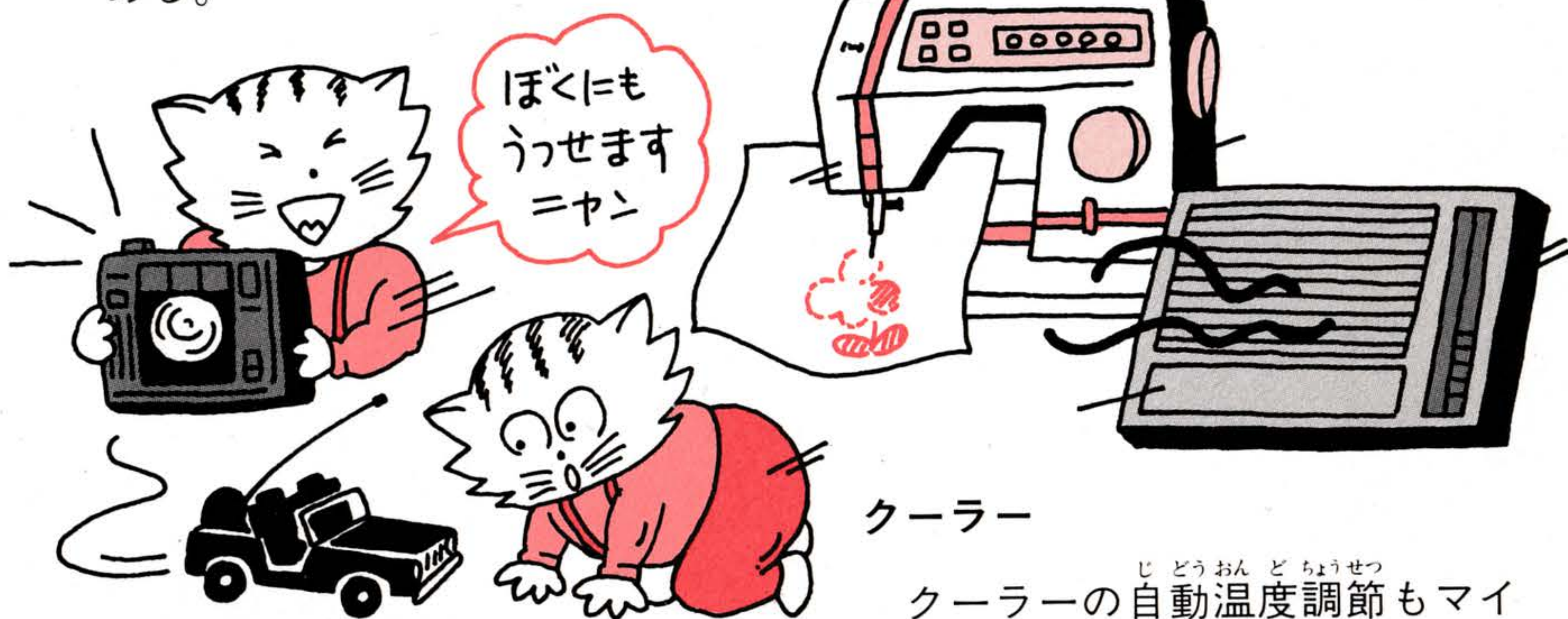
●身のまわりにあるコンピュータ

EEカメラ

レンズを^む向けてシャッターを^お押すだけでいいEEカメラ。距離を^{きょり}はかったり、明るさを^{あか}調べたりして、シャッター^{そくど}速度やしぼりを^き決める。

ミシン

^{はな}柄やいろいろな模様を自由^{もよう}自在^{じゆうじ}にしゅうしたりするのは、みんな^{ないぞう}内蔵コンピュータのはたらき。



自走プラモデル

たとえば戦車^{せんしゃ}やジープなどのプラモデル。どんなスピードで、どこで^ま曲がるかなど、みんな^{ないぞう}内蔵コンピュータしだいだ。

クーラー

クーラーの自動^{じどう}温度調節^{おん ど ちようせつ}もマイコンがやっているのだ。



それだけじゃないよ
まわりを^み見わたして
ごらん

せん^{せん}洗たく機^きだって

そうじ機^きだって

あた^{あた}ら^ら新しい電化製品^{でん か せいひん}や

じ^じどう^{どう}か^かき^き自動化機器^きには

ほとんど

マイコンが^{はい}入っている



楽しいディスプレイ



しごとの結果が目で見てわかる

パソコンにはディスプレイがつきものだ。ディスプレイはテレビのブラウン管^{かん おな}と同じで、そこにもじ^{もじ}を書いたり^か絵を描いたり^{え か}、いろいろなものを^{ひょうじ}表示する。

パソコンのできるしごとはたくさんあって、ひとくち^{くち}で説明^{せつめい}できないけど、ディスプレイに^{ひょうじ}表示するのも、たいせつなしごとのひとつなのだ。ディスプレイがなかったら、パソコンなんて、おもしろくもおかしくもないってことになりそうだ。

パソコンがはたらいた結果^{けっか}を表示^{ひょうじ}して見せるのがディスプレイだといっておこう。ディスプレイがあるから、しごとの結果^{けっか}がわかるのだ。

ディスプレイが、パソコンにとって、なくてはならないものだということがよくわかったはずだ。

ディスプレイの表示^{ひょうじ}のしかたにも、いろいろあるよ。パソコンに命令^{めいれい}するプログラムを表示^{ひょうじ}させたり、表^{ひょう}やグラフはもちろん、ゲームのキャラクターや、アニメーション、直線^{ちくせん}や曲線^{きくせん}を組み合わせ^{く あ}たグラフィックスなど、なんでも表示^{ひょうじ}してしまう。

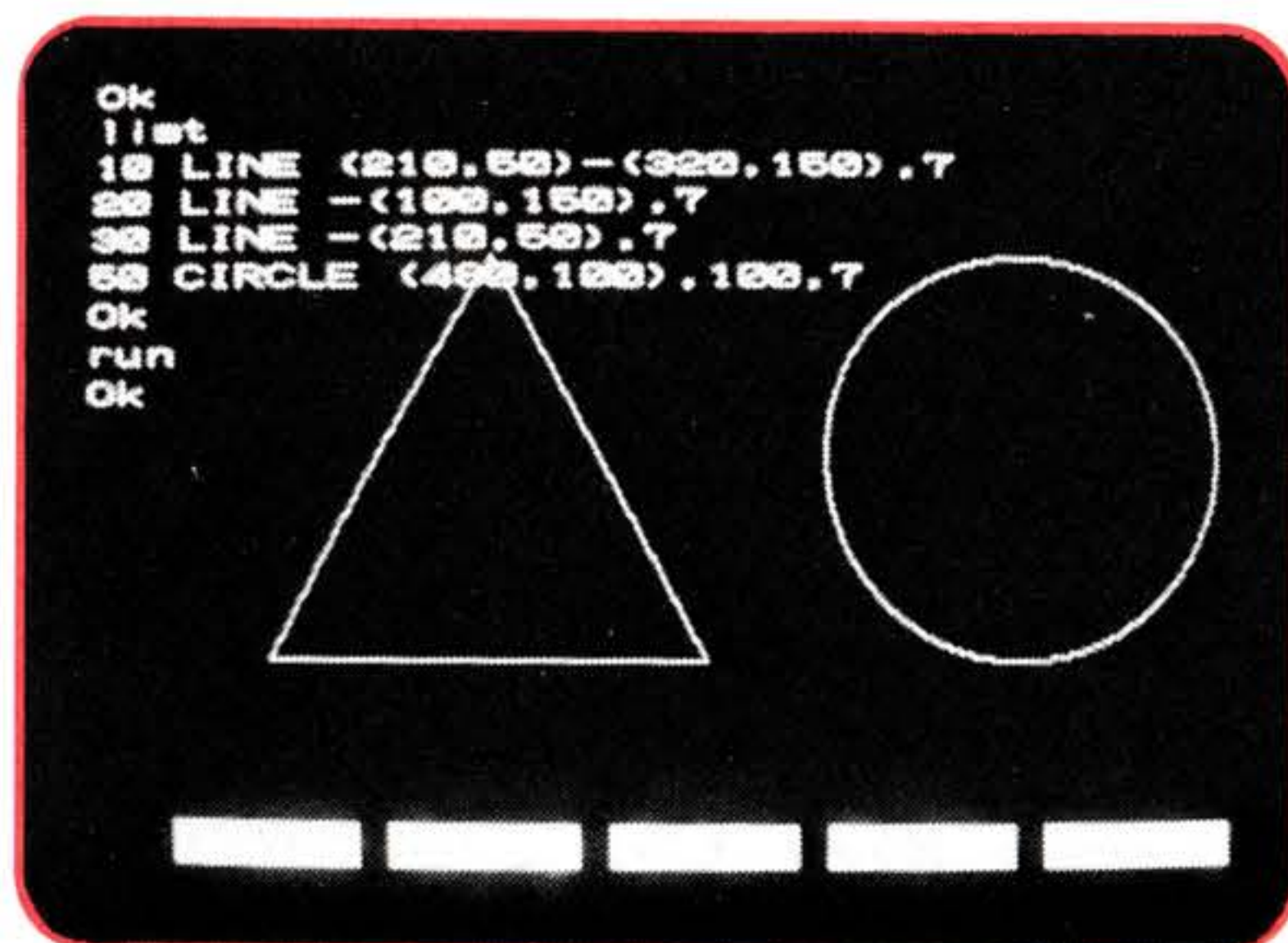
いろ
色は15色^{しよく}を使って
ひょうじ
表示^{ひょうじ}できるから
がめん
画面はカラフル
いろ
色いっぱいだよ



ディスプレイは
せんよう
専用ディスプレイの
ほかに
かていよう
家庭用テレビも
つか
使えるのだ

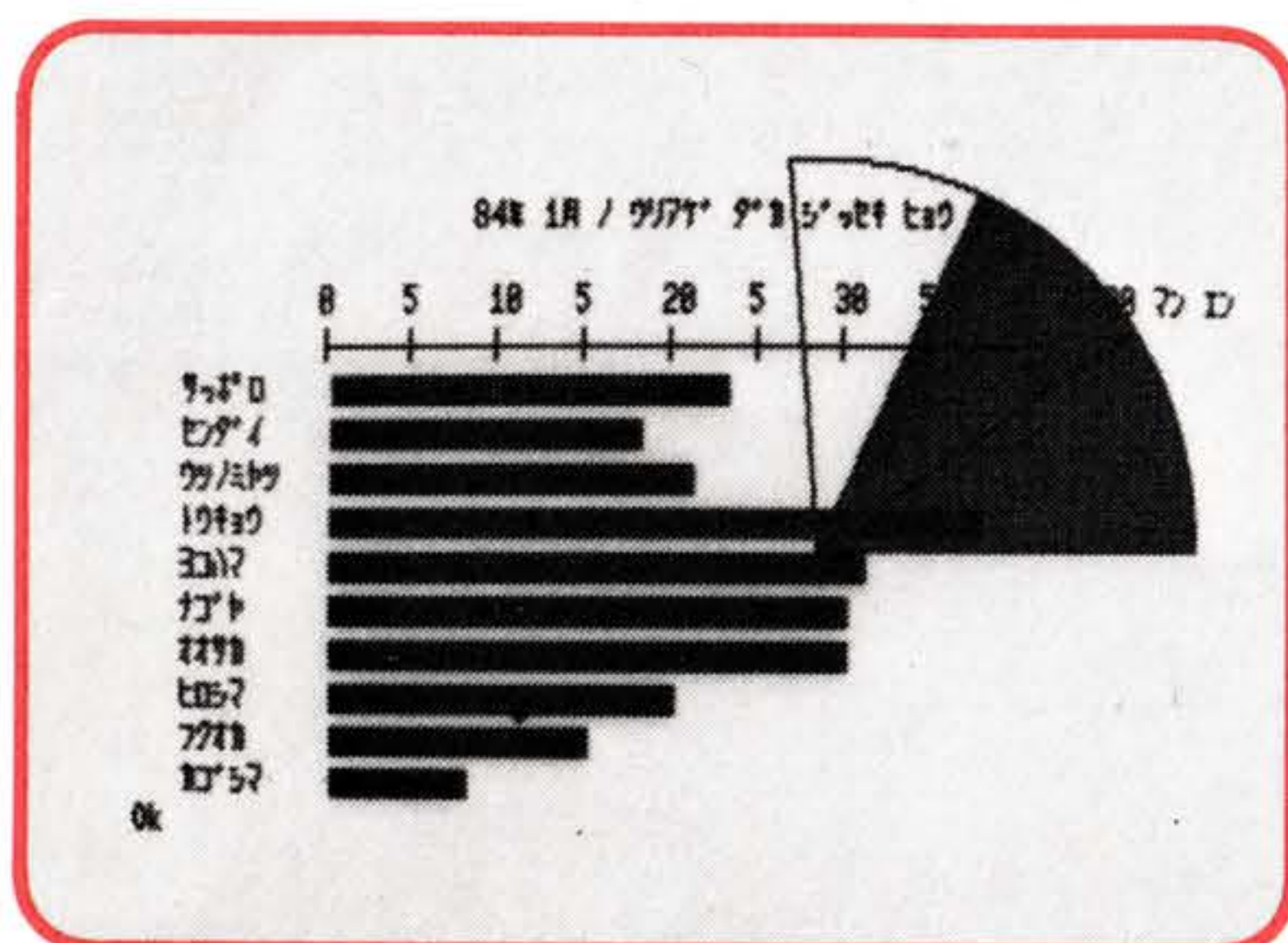


●パソコン画面表示のいろいろ

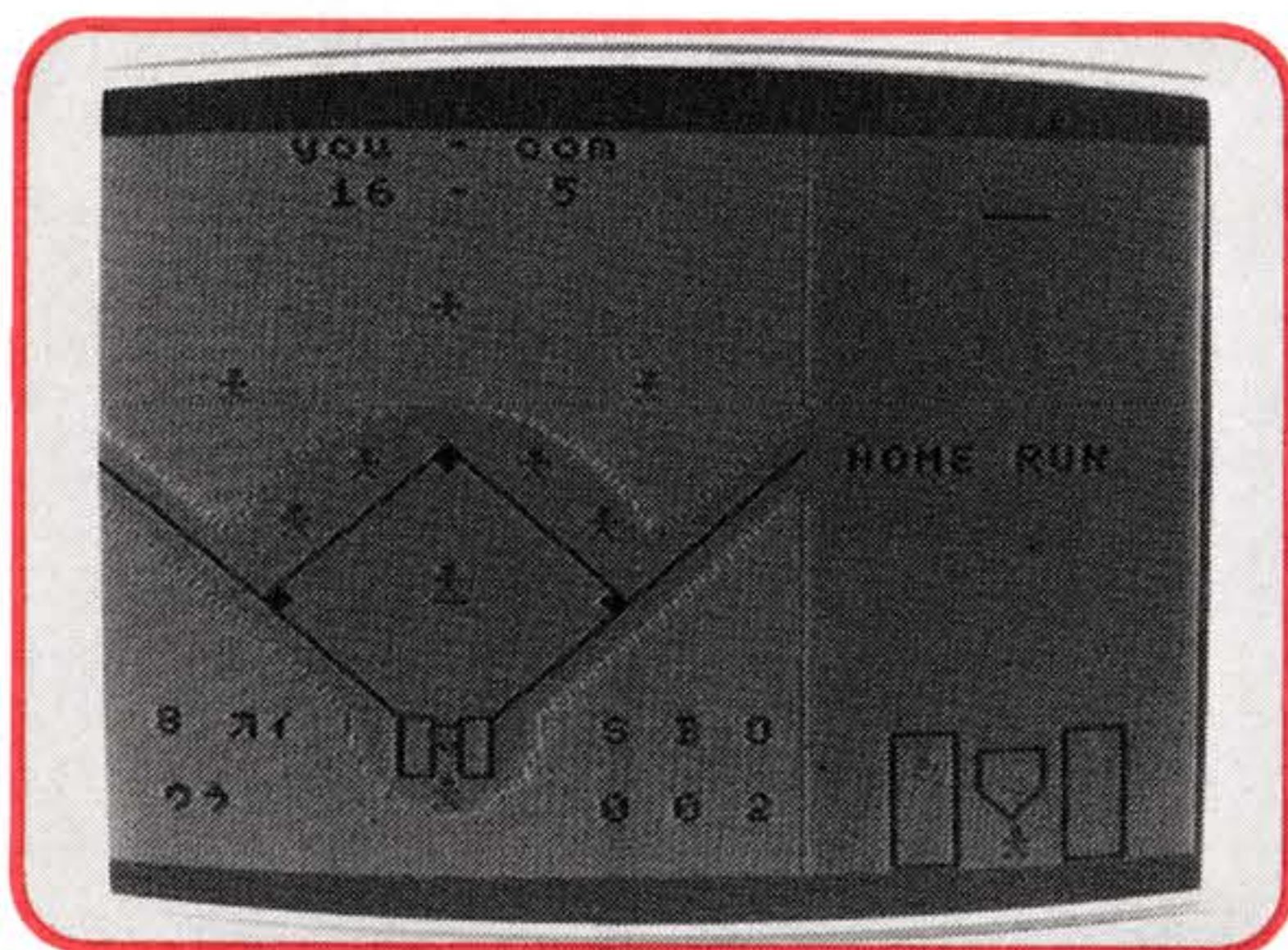


入力したプログラムの表示。まちがいないかなどをたしかめられる。

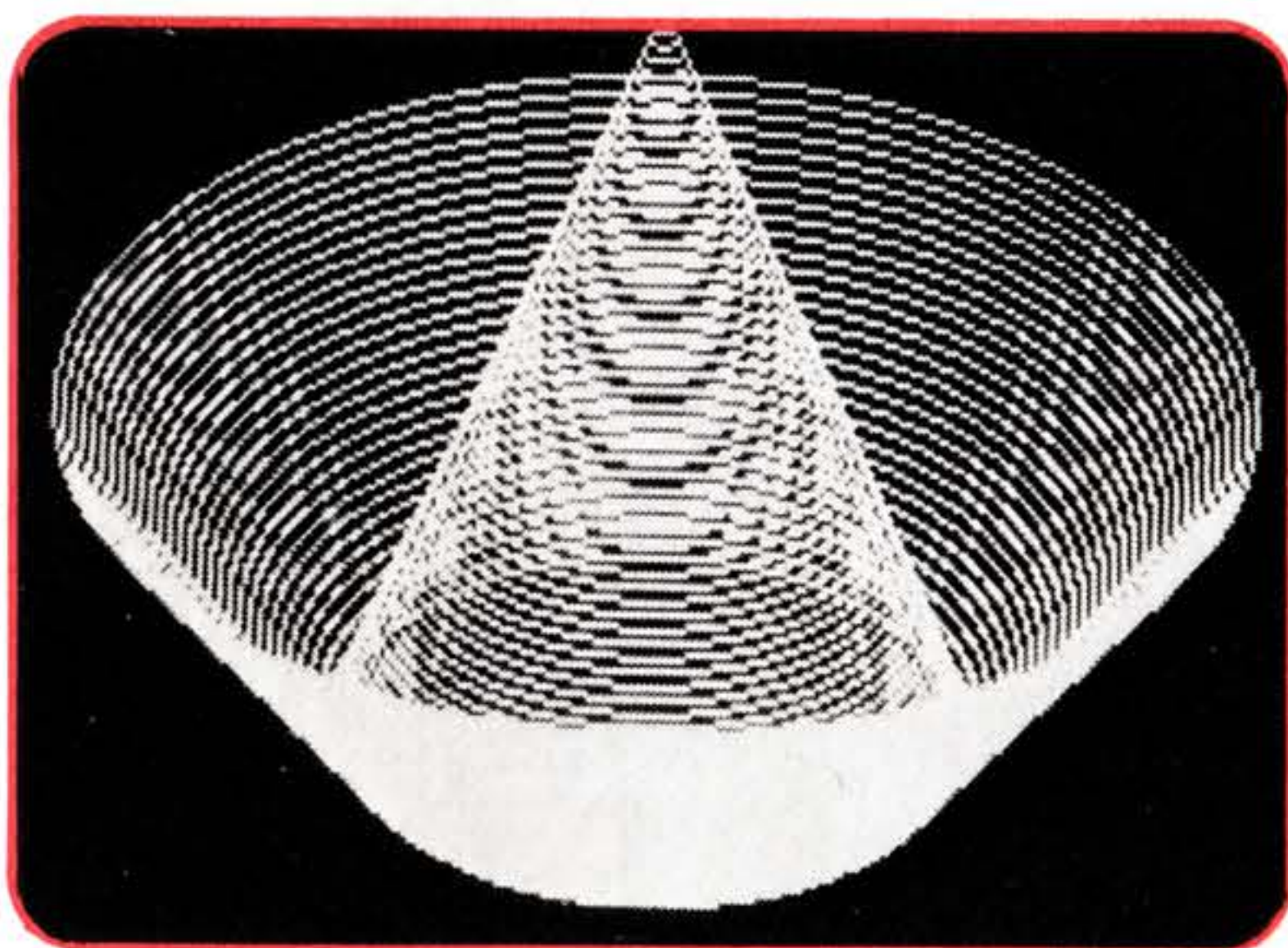
表示画面には
グラフィック画面と
テキスト画面があるよ



文字を表示するのが
テキスト画面なんだね



円グラフや棒グラフも自由自在。
しごとにも勉強にも活用できる。



なんといっても楽しいのが、パソコンゲームの画面だ。ちょこまか動くキャラクターが、またかわいいよ。

カラフルに、ダイナミックに、
そうぞうしく作ってみよう。

カラーグラフィックスは、手でデザインするより、パソコンにやらせたほうが、思いもしなかった効果が出たりして、楽しい。

④

パソコンって どんなマシンなの？



キーボードやテレビやテープや

パソコンは、どんな機器があれば、パソコンといえるのだろう。

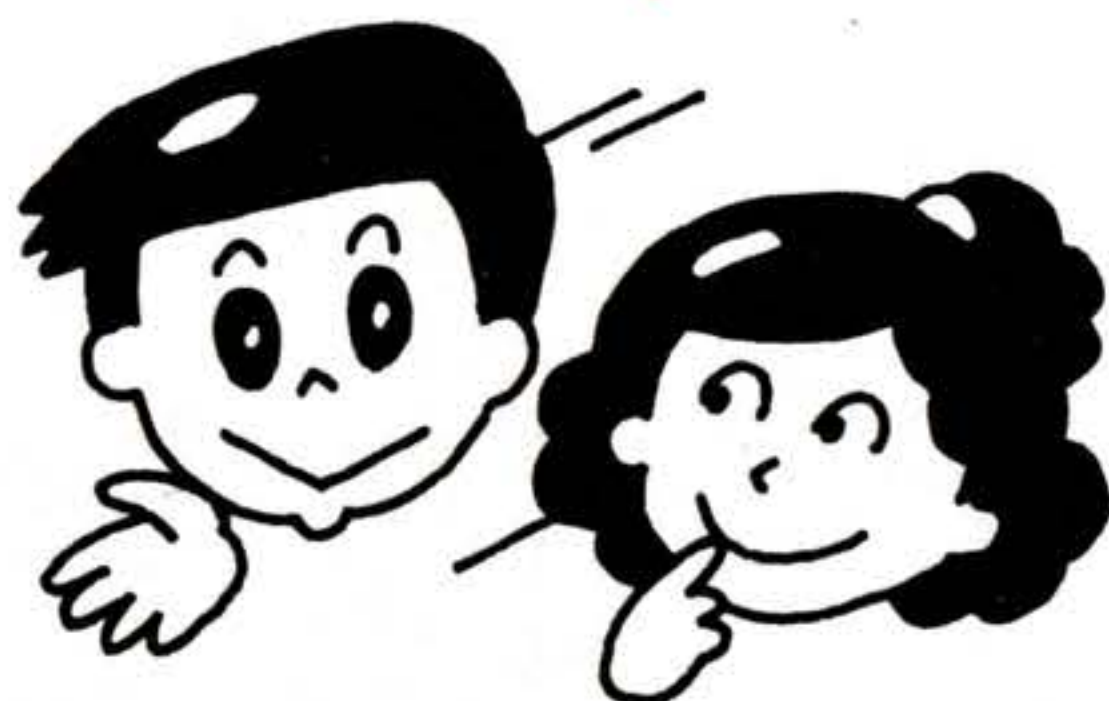
まず、本体。命令を記憶したり、命令を実行したりするパソコンの頭脳だ。タイプライターみたいに、文字や記号を打ち込むキーボードが付いているからすぐわかる。本体・キーボードともいう。

本体・キーボードがあれば、命令を入力できし、それを記憶したり計算して実行したりできるけど、なにかまだものたりない。

実行した結果を知るためには、それを表示するテレビやディスプレイがほしい。結果を長く保存するには、紙に印刷しておくためのプリンタも必要だ。

こうした本体・キーボード以外の付属機器を周辺機器といっている。このほか、周辺機器の主なものには、カセットテープ、テープレコーダ、ディスク装置とディスケット。ほかにもまだまだあるが、これだけそろっていれば、まあ、いちにんまへのパソコンといっていいいだろう。

テープは
音楽を聞く
ふつうのテープで
いいんだよ

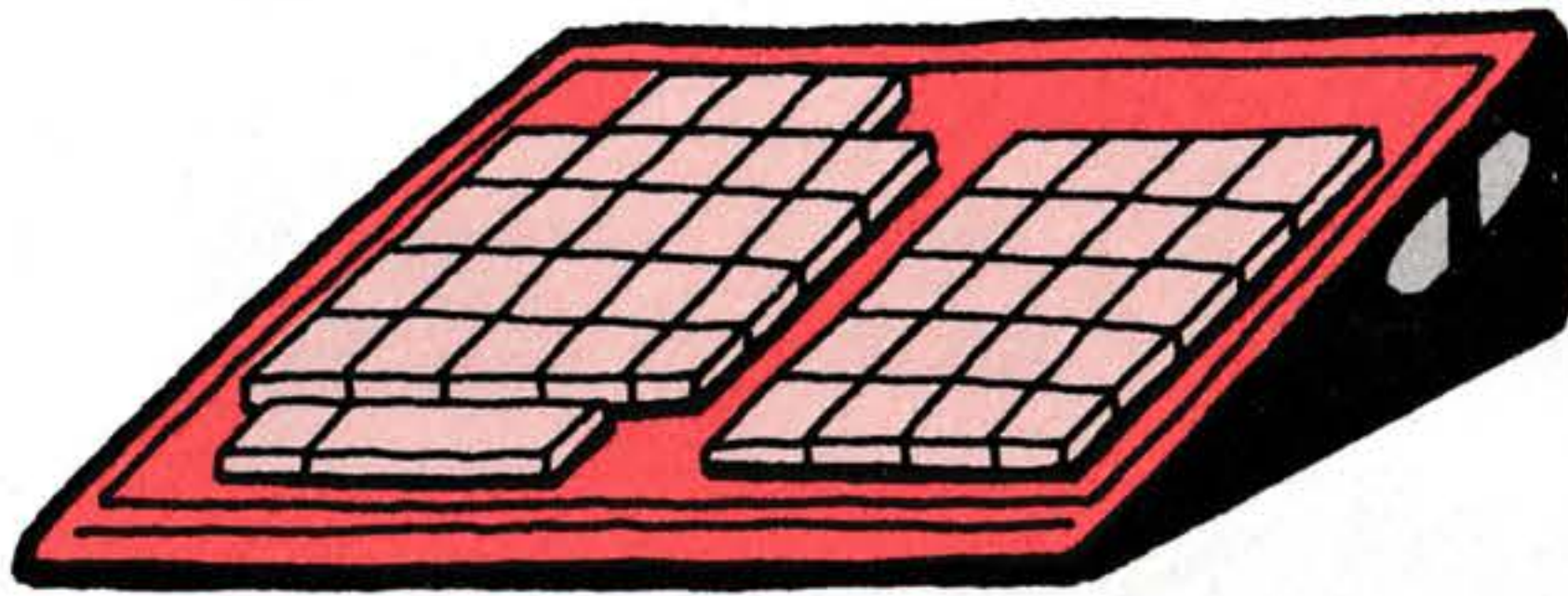


音楽を録音するように
テープに
プログラムという
命令を
記憶させておくのだ

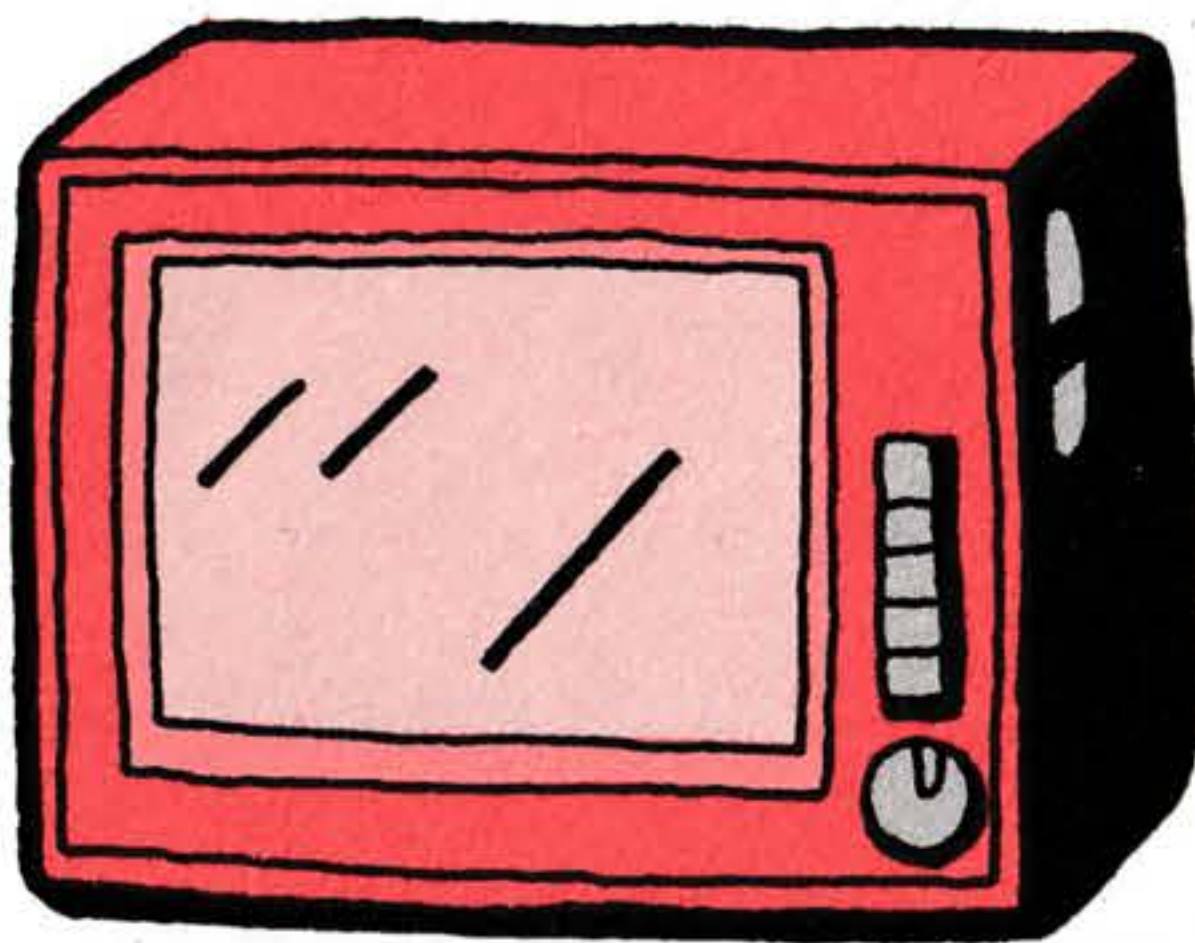


●パソコンの構成品

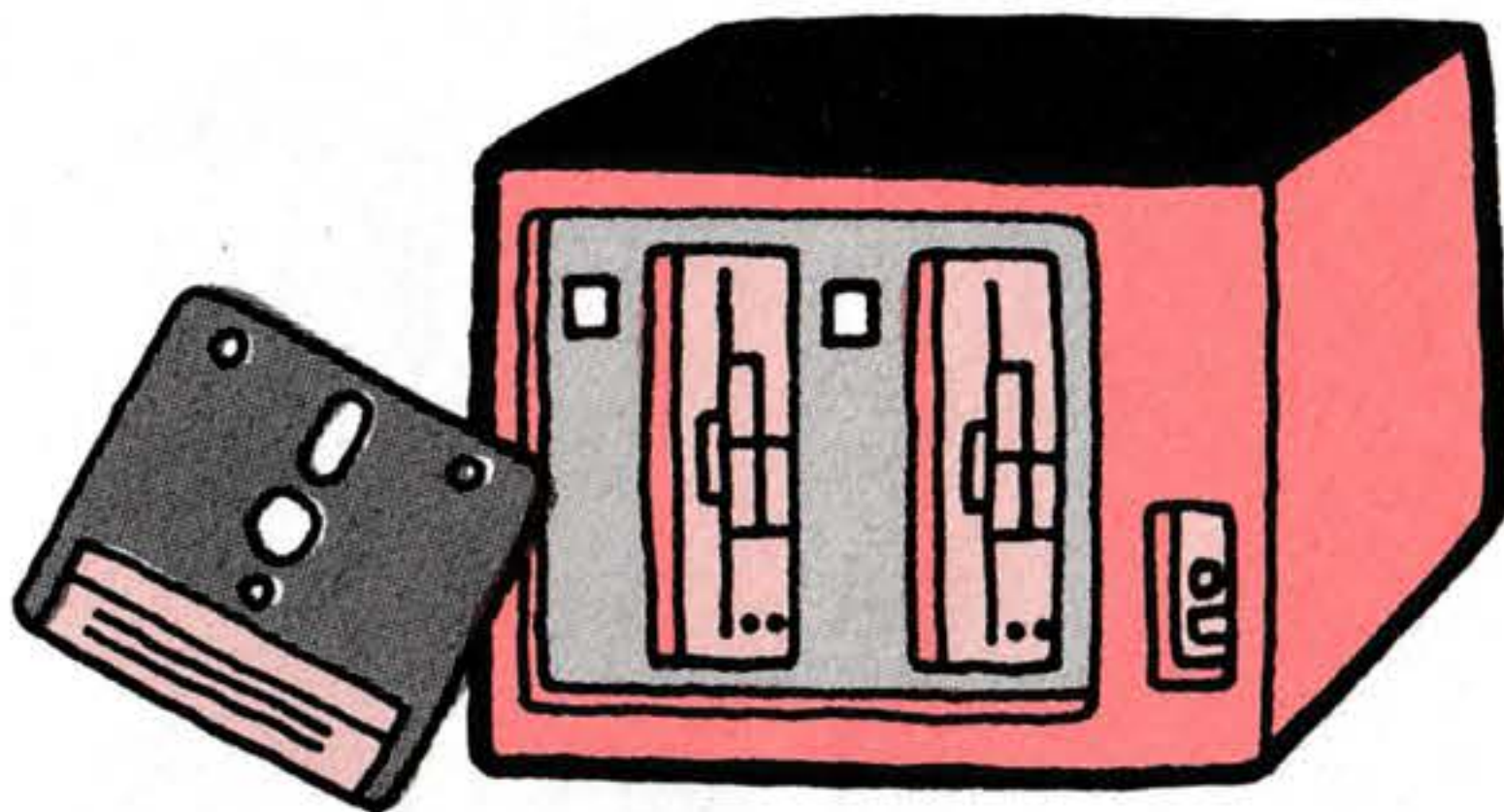
本体・キーボード



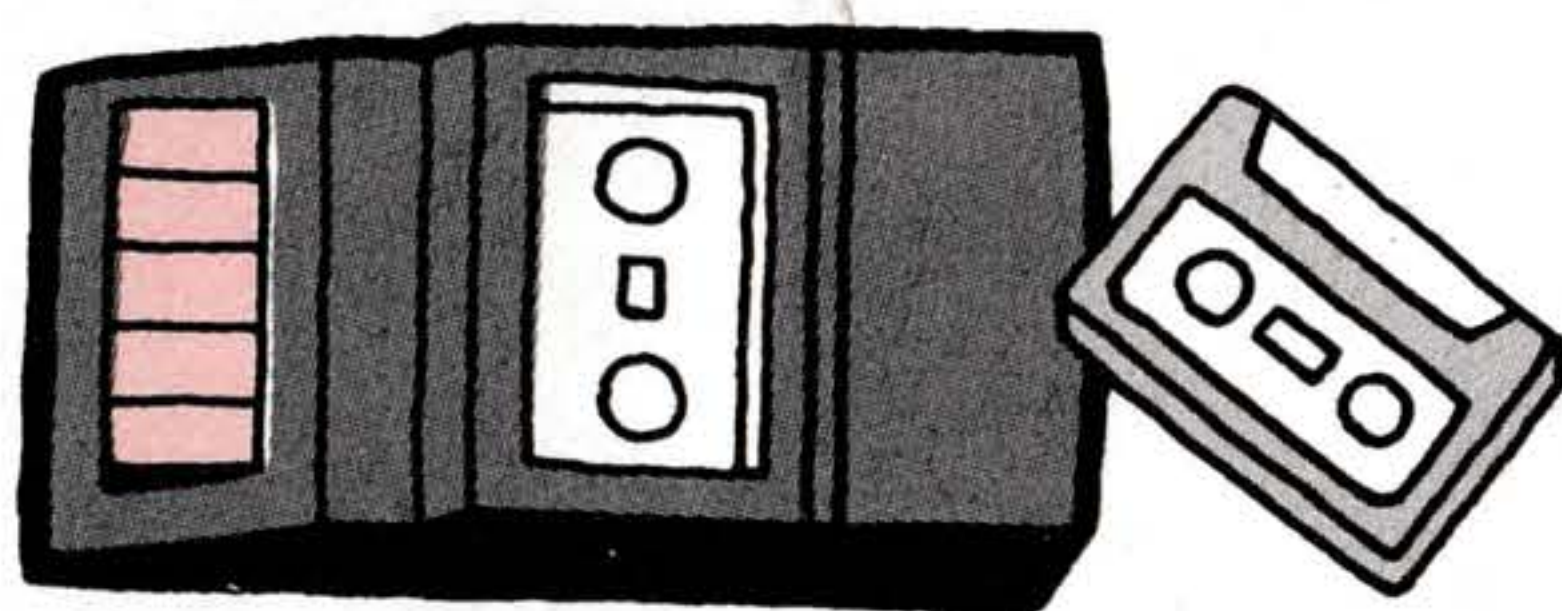
テレビ（ディスプレイ）



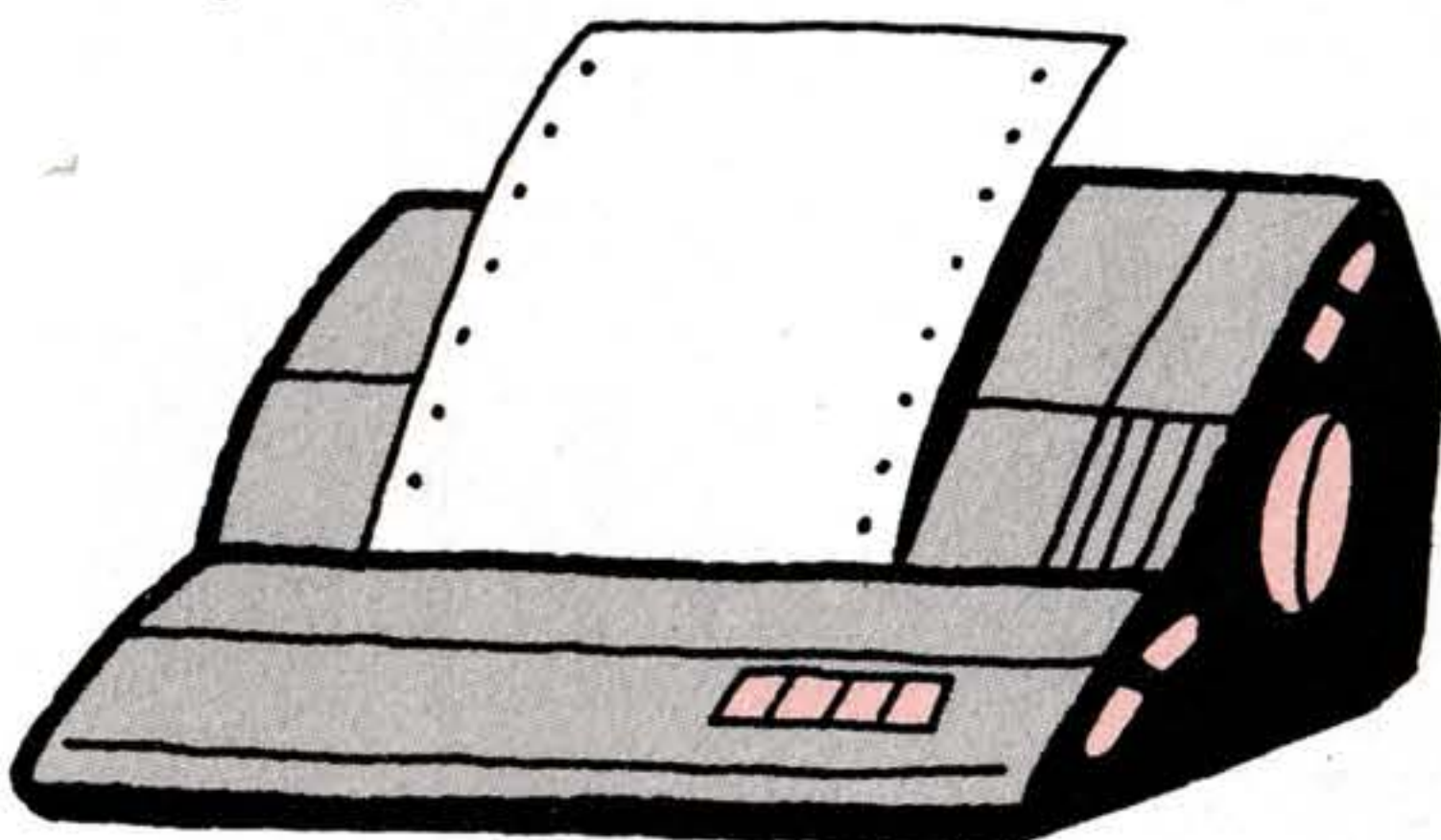
フロッピーディスク



テープレコーダ



プリンタ



本体・キーボード

キーボードは文字や記号などを
本体に入力する入力装置。本体は
ここから入力された命令（プログラ
ム）の記憶と実行をおこなう。

テレビ（ディスプレイ）

入力したプログラムを表示した
り、本体が実行した計算結果や絵
や図などを表示させる。出力装置
のひとつ。

フロッピーディスク

一度作ったプログラムを記憶さ
せておき、必要なときにここから
呼び出して使用する。外部記憶装
置という。

カセットテープとテープレコーダ

外部記憶装置のひとつで、はた
らきはフロッピーディスクと同じ。
手軽だがスピードがやや遅い。

プリンタ

プログラムや本体が実行した結
果を保存するために、紙に印刷す
る。出力装置のひとつ。

5

パソコンのしごと



にゅうりよく き おく えん ざん せい ぎょ しゅつりよく
入力、記憶、演算、制御、出力

コンピュータにはいったいなにができるのだろう。コンピュータのしごとについて考えてみよう。

あたえられたプログラムとデータにしたがって、しごとをするわけだが、しごとの内容は大きく分けて5種類ある。

入力、記憶、演算、制御、出力が、コンピュータのしごとだ。大きなコンピュータも、パソコンぐらいの小型のコンピュータも、できるしごとの内容は同じだ。

●パソコンのしごとの種類

入力……プログラムやデータを外部から本体の記憶装置に入れるしごと。キーボードやジョイスティックが入力装置だ。

記憶……入力したプログラムやデータを記憶しておくしごと。本体内部の記憶装置のほかに、フロッピーディスクやカセットテープなどの外部記憶装置がある。

演算……記憶装置に記憶したデータを使い、プログラムの命令どおりに、計算や処理

プログラムは
 コンピュータに与える
 しごとの内容を書いた
 命令書なのだ



本もののコンピュータは
 5つの要素を
 そなえていれば
 いいんだな

パート0 コンピュータってな～に？

の^{じっこう}実行をするしごと。本体内部にある^{ほんたいないぶ}中央演算処理装置（CPU）がおこなう。

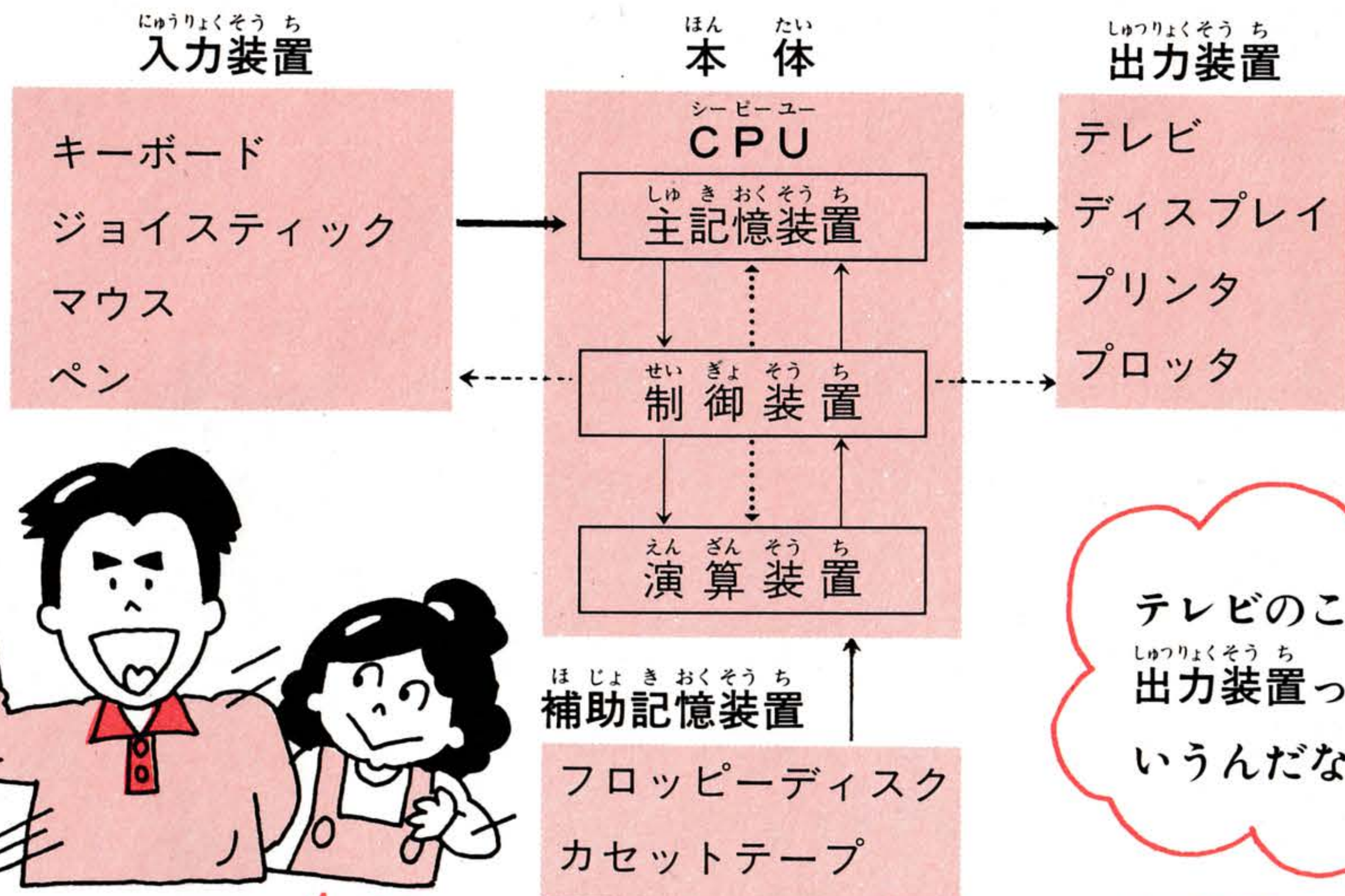
制御……コンピュータを^く組み立てているすべての装置が、それぞれしごとの^{はんい}範囲を守って^{ただ}正しく^{どうさ}動作するようにコントロールするしごと。

出力……演算装置の実行結果を外部に^{しゅつりょく}出力して知らせるしごと。テレビ（ディスプレイ）やプリンタが^{しゅつりょくそうち}出力装置である。

ものおぼえがよくて
計算が早いから
パソコンはやっぱり
さんすうの天才だ

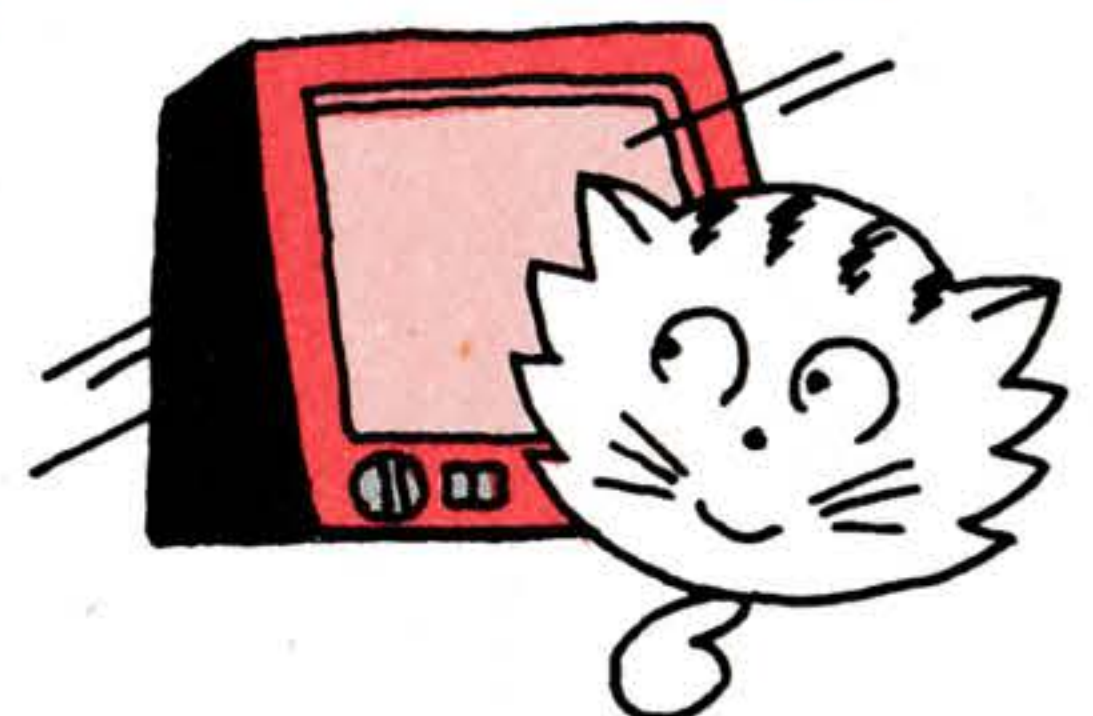


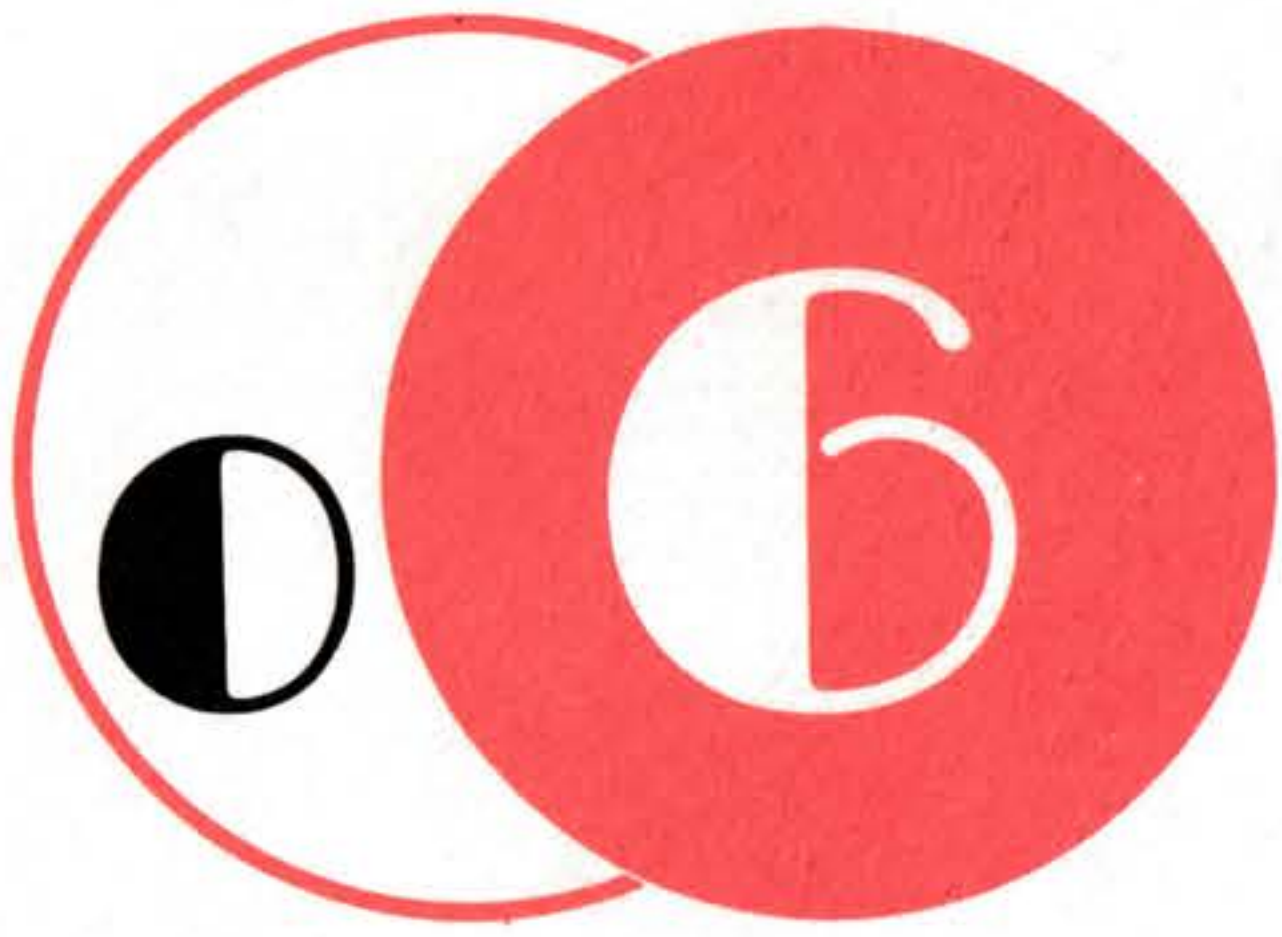
●パソコンのシステム



ジョイスティックをゲームに使えばおもしろさバツグンだ
キャラクタや駒の移動が思いのままなもの

テレビのことを
出力装置って
いうんだな





パソコンは 0か1だけの世界



でんりゅう 電流があるかないかだけで判断

パソコンと長くつき合って行こうとするなら、
少しむずかしいかもしれないけれど、2進法って
いう数字の数え方も知っておいたほうがいい。

ふつうさんすうなどで使っている数え方は、9
の次に桁上がりして10になるから、10進法だ。そ
れくらいは知っているよね。

ところが、2進法は0と1の2種類の数字しか
使えないから、0、1ときて、次はすぐ桁上がり
して10となるのだ。

なぜ、そんなめんどろな数え方をしなければな
らないかって？ それは、コンピュータの世界が、
0と1しかない世界だからだ。

パソコンが情報を判断する材料は、電気回路に
電流がきていないか、きているかだけだ。それで
電流がきていなければ0、きていれば1と表現す
る。すべての情報を0と1の組み合わせで表現す
るから、どうしても2進法でなければならないと
いうわけ。パソコンは、電流があるか、ないかだ
けで判断するというのが、たいせつなのだ。



じかん 時間の数え方は
びょう 60秒で1分
ぶん 60分で1時間と
けたあ 桁上がりするから
しんほう 60進法なのだな

●10進数と2進数

10進数と2進数の関係

10進数	2進数
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
100	1100100



10進数は0から9まであるけど
2進数の数字は0と1だけなのだ



ON

1

OFF

0



10進数はデシマル、2進数はバイナリ

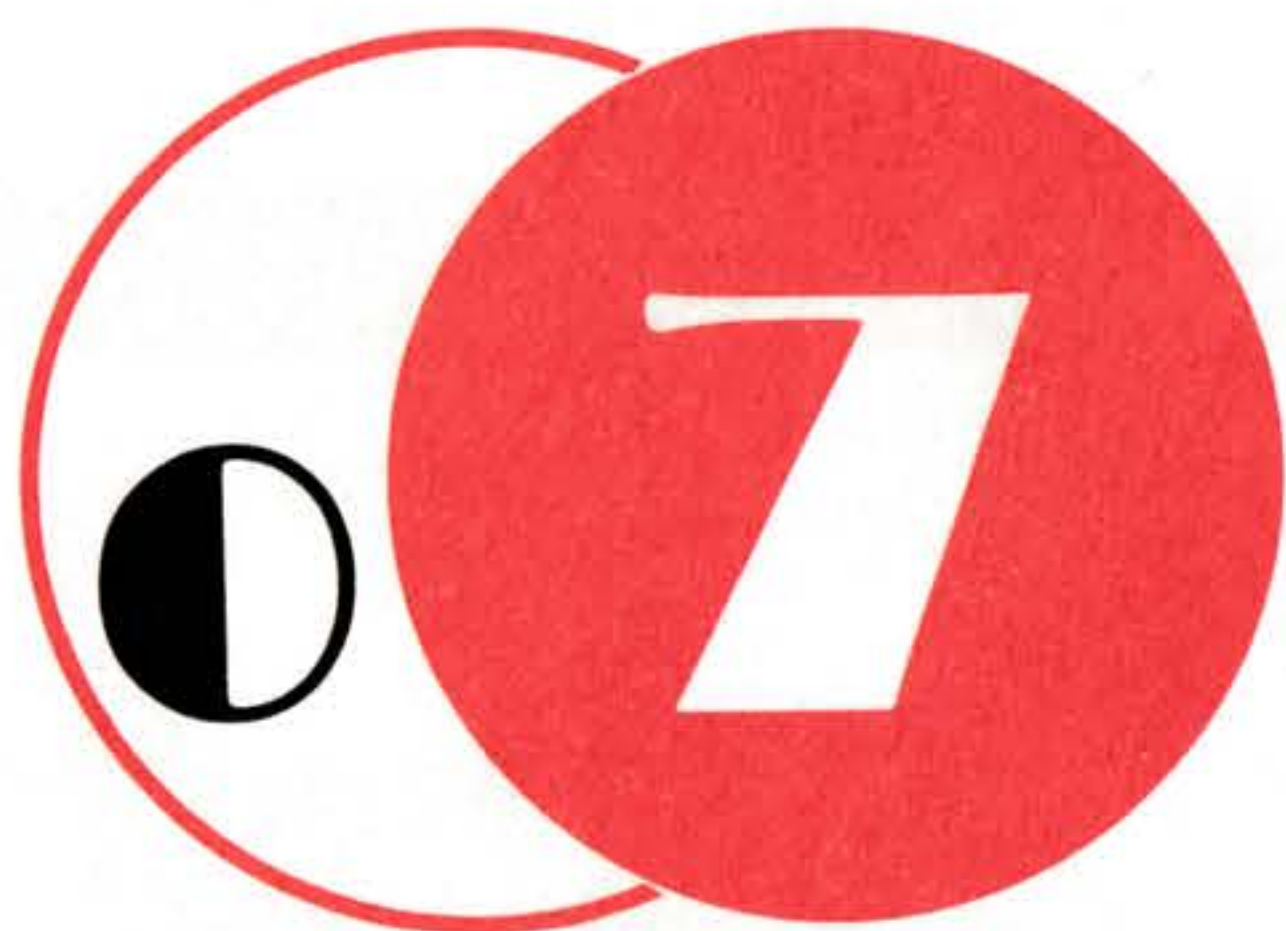
16進数はヘキサといって、数のあとにHをつける

16進数は9までが10進数と同じ

10から15までがA、B、C、D、E、Fだ

パソコンの内部は電流ONとOFFだけの世界だから、ONは1、OFFは0として、2進法だけが動いている。だからといって、それを紙に書いていくときなどは、2進法で書くと桁数ばかり多くなってめんどろだ、というわけで、16進法で書かれることがある。

たとえば、10進数の15は2進数では1111と4桁だが、16進数ではただ1字のFですむからだ。



ビットとバイト



かくのう じょうほう けたすう
ビットは格納する情報の桁数だ

8ビットパソコンとか、メモリ容量^{ようりょう}1000^{ケー}Kバイトとかいうように、「ビット」と「バイト」はよくきくことばだが、いったいなんだろう？

パソコンの^{じょうほう}情報が、すべて2^{しんすう}進数の0と1でできているのは、もう^{まえ}前に^か書いたから^し知っているね。

ビットは、この2^{しんすう}進数の^{じょうほう}情報の^{けたすう}桁数のことだ。

2^{しんすう}進数の^{じょうほう}情報が^{はい}入っている^{ようりょう}容量をメモリ（^き記憶^{おく}装置）^{そうち}）というのだが、そこに^{かくのう}格納する^{ようりょう}容量の^{たんい}単位がビットだと^{かんが}考えていい。

メモリの^{おお}大きさの^{たんい}単位がビットだが、この8ビットを^{あた}新しい^{たんい}単位として1バイトという。なぜだろう？ メモリの^{ようりょう}容量などをビットで^{かず}数えると^{かず}数が^{おお}大きくなってしまふのと、8ビットCPUとか16ビットパソコンなどというように、8ビットを^{たんい}単位として^{かんが}考えると、なにかとべんりだからなのだ。

そのほか、^{じょうほうりょう}情報量の^{たんい}単位としては、^{ケー}Kバイト、^{メガ}Mバイト、^{ギガ}Gバイトもあるよ。

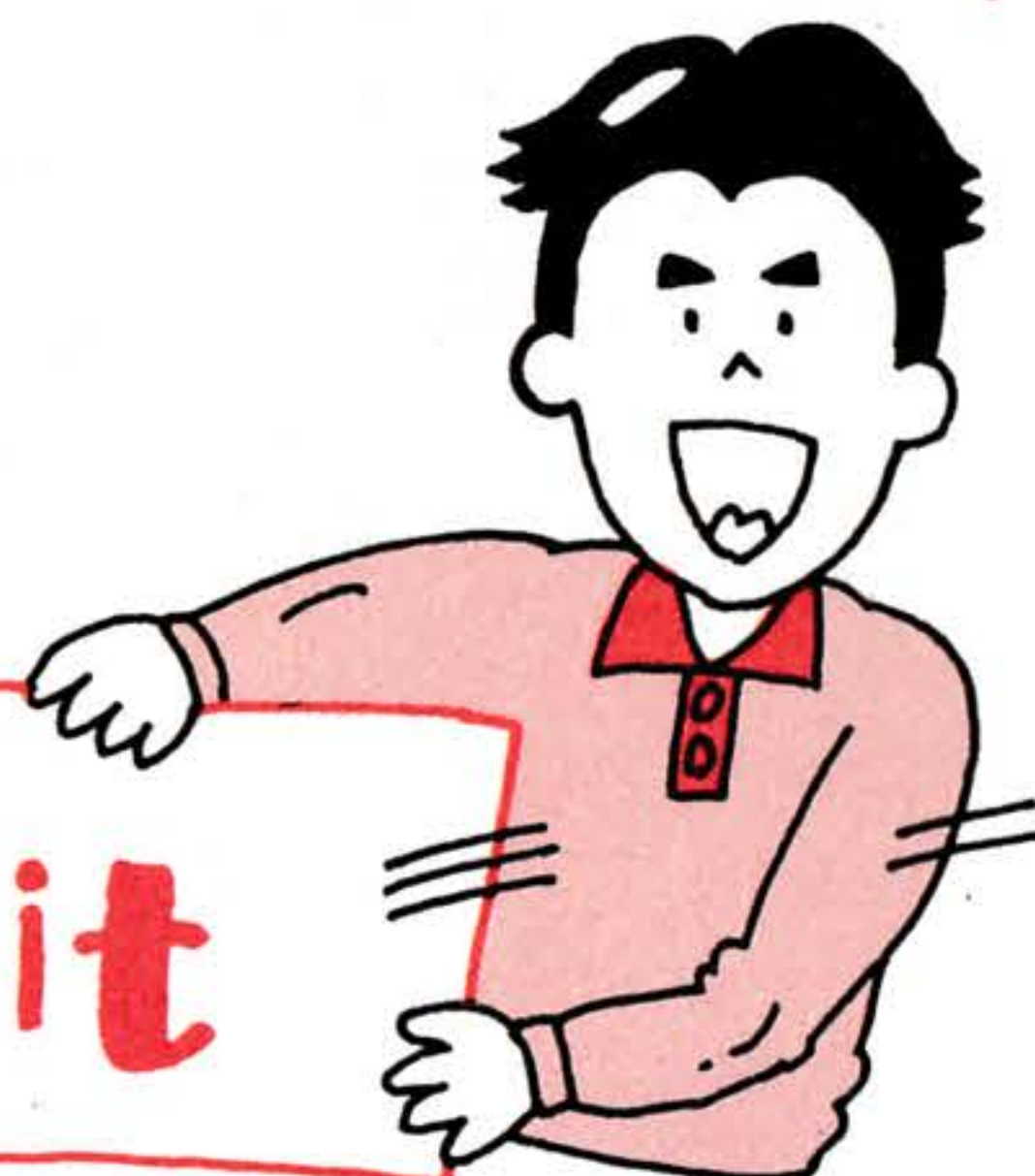
01なら2ビット

0001なら

4ビットなのだ



ビット (Bit) とは
^{しんすう}2進数 (Binary digit)
^{りやく}の略で^{けたすう}桁数のことだ



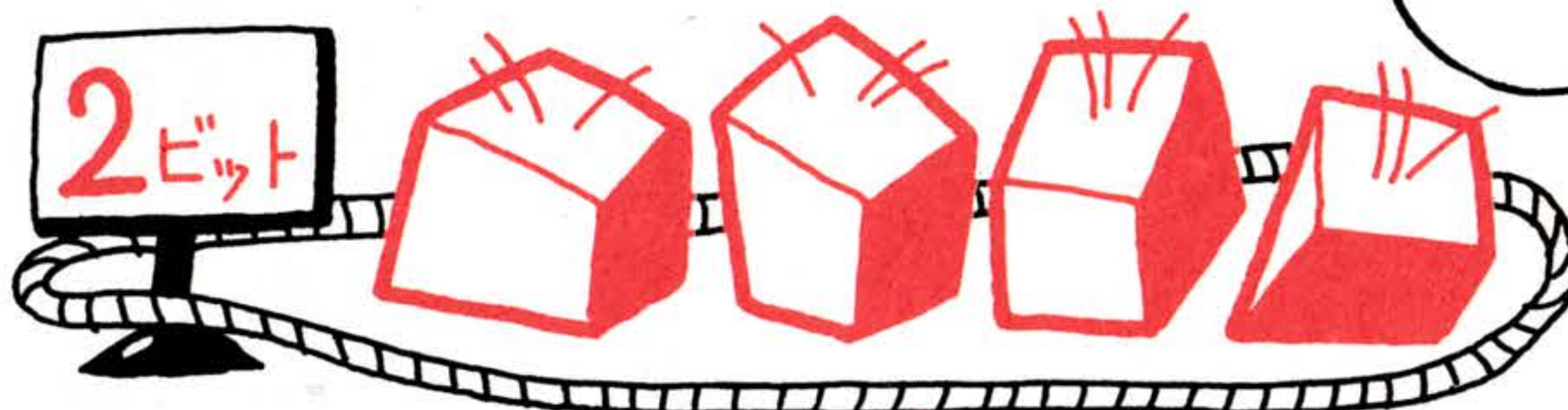
●各ビットの情報量

かく じょうほう
各ビットの情報

1 ビットの情報	0, 1
2 ビットの情報	00, 01, 10, 11
3 ビットの情報	000, 001, 010, 011, 100, 101, 110, 111
4 ビットの情報	0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111



1 ビット増加するごとに
情報量が2倍になる
バイバイゲームだ



じょうほう おお しめ たん い
情報の大きさを示す単位

1 ビット = 2 進数 1 桁の情報量

1 バイト = 8 ビット

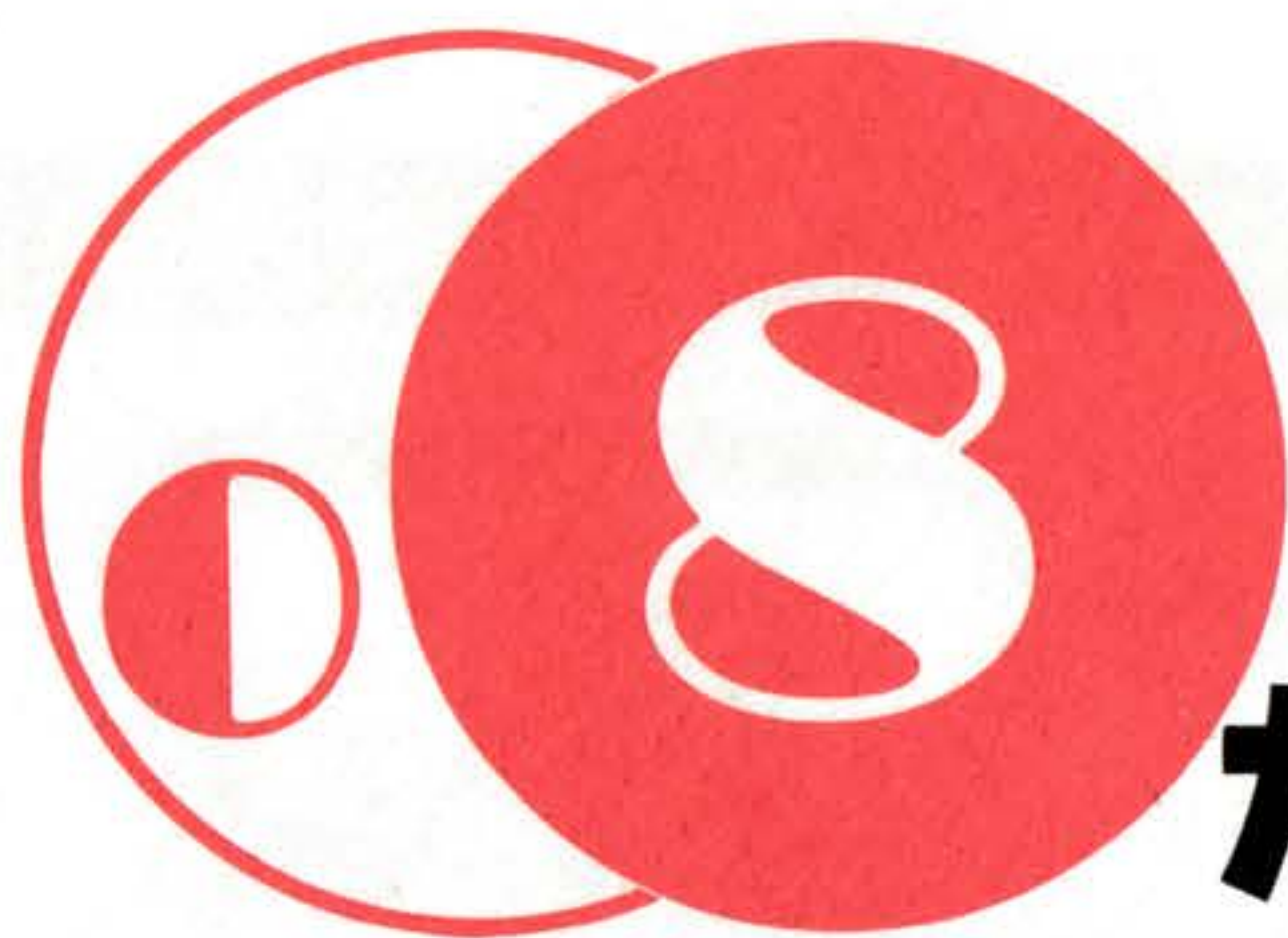
1 K バイト = 1024 バイト

1 M バイト = 1024 K バイト

1 G バイト = 1024 M バイト

1 ビットの情報
2 進数 1 桁だから
1 か 0 の 2 通り
2 ビットなら
00, 01, 10, 11 の
4 通りあるよ





ぼくのパソコンだ たいせつに扱ってね



せつち ばしょ かんきょう き くば 設置する場所の環境に気を配って

ねんがん 念願がかなって、ようやく買ってもらったパソコンだ。たいせつに扱ってね。あまり乱暴に扱^{あつか}うと、パソコンは、きみのいうことをきかなくなってしまうよ。

キーボードのキーは、多少強^{たしやうつよ}くたたいても、続^{つづ}けて打ち込^うんでも、こわれる心配^{しんぱい}はないから、どんどんキータッチの練習^{れんしゅう}をしてもいいけれど、外^{がい}気^きなどの環境^{かんきょう}がわるいと、パソコンはちょっと弱^{よわ}いところがあるから、気^きをつけてね。

だからといって、パソコンは、ほかの精密機械^{せいみつ きかい}にくらべて、もろいのかというと、決^{けつ}してそんなことはない。むしろ、ほかの機械^{きかい}よりも構造的^{こうぞうてき}には強^{つよ}くできているから、あまり神経質^{しんけいしつ}になることはない。

でもやはり、精密機械^{せいみつ きかい}なのだから、注意^{ちゅうい}するにこしたことはないだろう。これから長^{なが}くつき合^あっていくのだから、たいせつに扱^{あつか}って、かわいがってやろう。とくに、振動^{しんどう}や熱^{ねつ}、ほこり、たばこの煙^{けむり}などはパソコンの大敵^{たいてき}なのだ。

そう さ まえ
操作の前にちょっと待て
あつか かた こころ え
扱い方の心得を
よくおぼえてからGO!



でもビビったりしない
キーボードからは
う こんどん 打ち込もう

●パソコンの大敵は？

1. 常時振動する場所に置かない

精密な電子部品がプリント基板にパターン配線してあるから、振動には弱いところがある。常時振動する場所には、置かないことだ。



2. 衝撃を与えない

ぶついたり、落っこしたりは大敵だ。持ち運びのときは十分気をつけよう。



3. 直射日光にさらさない

パソコンは急激な温度変化に弱い。とくに、内部温度が急上昇しないようにする。直射日光にさらすなどは、もってのほかだ。



4. 煙やほこりを避ける

操作中にたばこを吸ったり、カバーをかけずに部屋をそうじしたりしない。煙やほこりが、すき間から忍び込んで、故障の原因になってしまう。



5. 電子機器の雑音に気をつける

電子機器から出る電磁波（ノイズ）の影響で、画面がぶれたり、色調が乱れたりするから注意。

よごれたからといって
水で洗ったりはするな
かわいた布で
軽くふくぐらいでいい





正しく接続しよう



かく き しゆ せつ めい しよ よ
各機種ごとに説明書をよく読んで

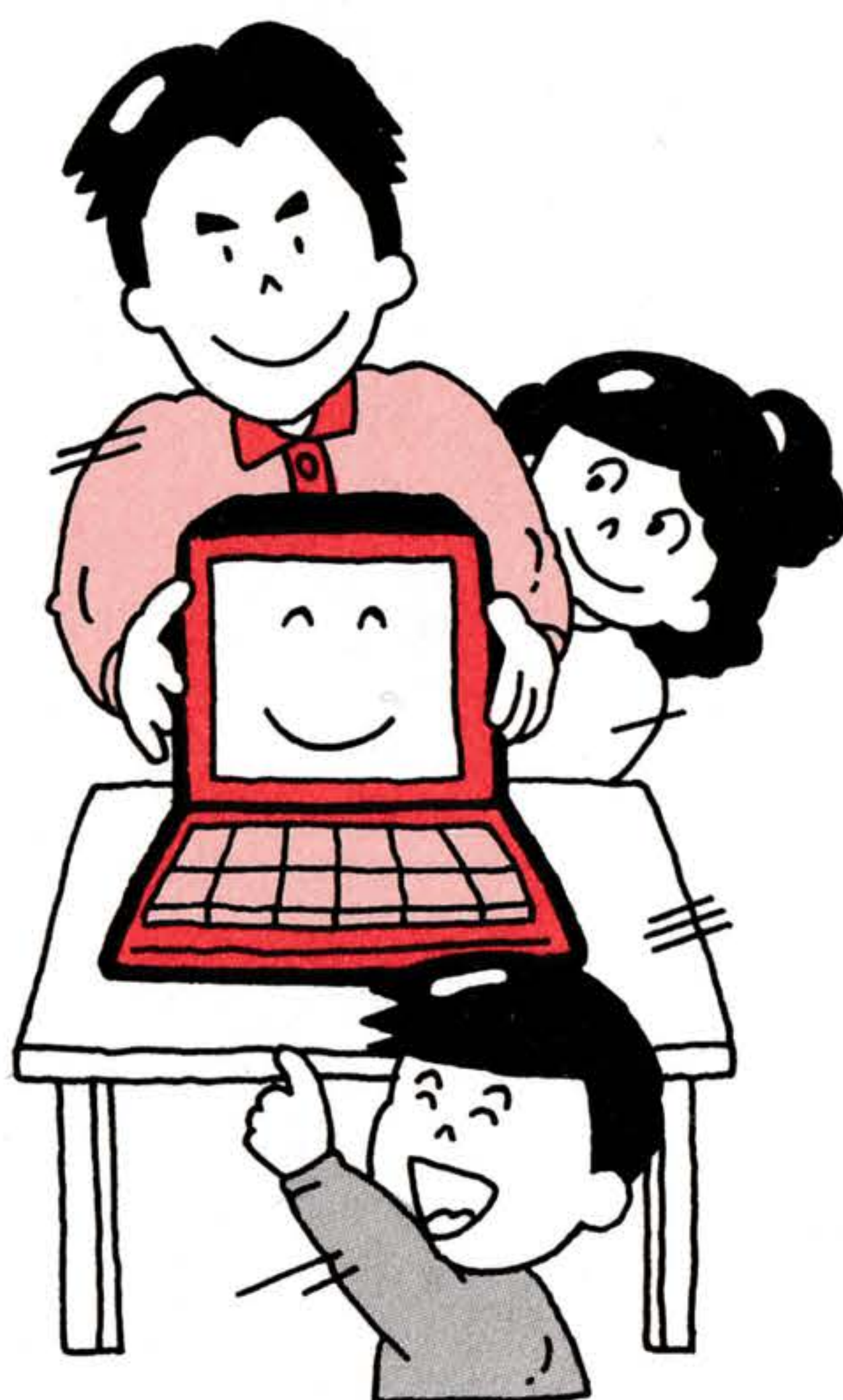
パソコンを使用する前に、設置する場所を決めよう。設置したあとで、しょっちゅう移動させなくてもいいように、場所選びを慎重にしよう。

場所が決まったら、本体・キーボードと周辺機器のセッティングだ。

本体・キーボードとテレビ（ディスプレイ）、プリンタ、カセットレコーダ、ジョイスティックなどの周辺機器を接続しなければならない。

接続の方法は、パソコンの各機種ごとに、多少違う場合があるから、パソコンを買ったときについてくる取扱説明書などをよく読んで、正しく接続することだ。

たとえば、MSXタイプなら、専用ディスプレイのほかに、家庭用テレビとも接続して使用できる。プリンタも、MSX専用タイプ以外のものを接続すると、グラフィックパターンが異なる場合があるなど、十分に知ってから接続しなければならない。次のページで、一般的な接続の方法を説明しておこう。



ディスプレイは
専用テレビか
家庭用テレビかで
接続方法も違うよ



●周辺機器の接続

テレビ（ディスプレイ）

の接続

家庭用テレビにも接続できるMSXには3通りの接続方法がある。

1. 家庭用テレビのアンテナ端子に、RF

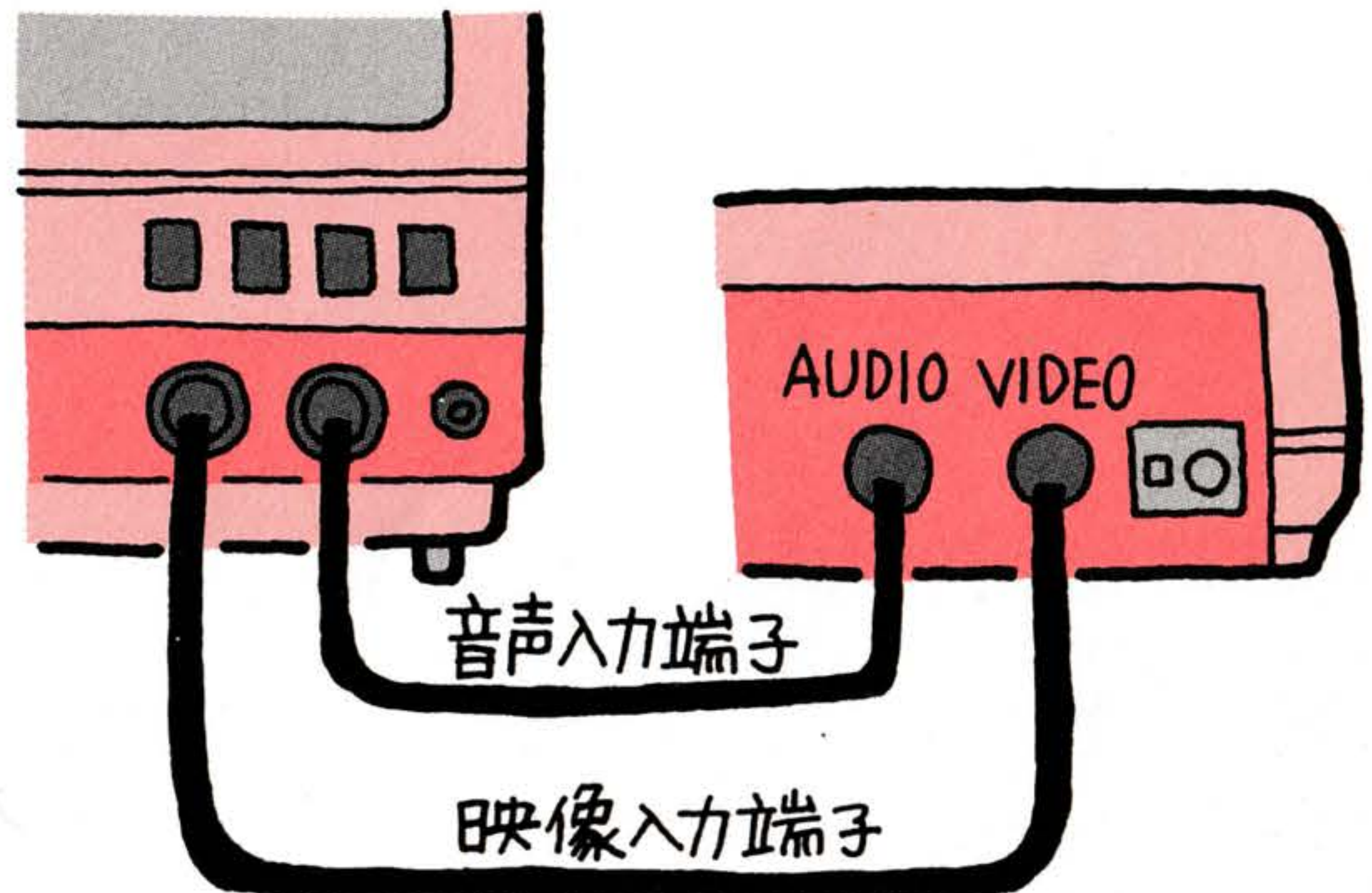
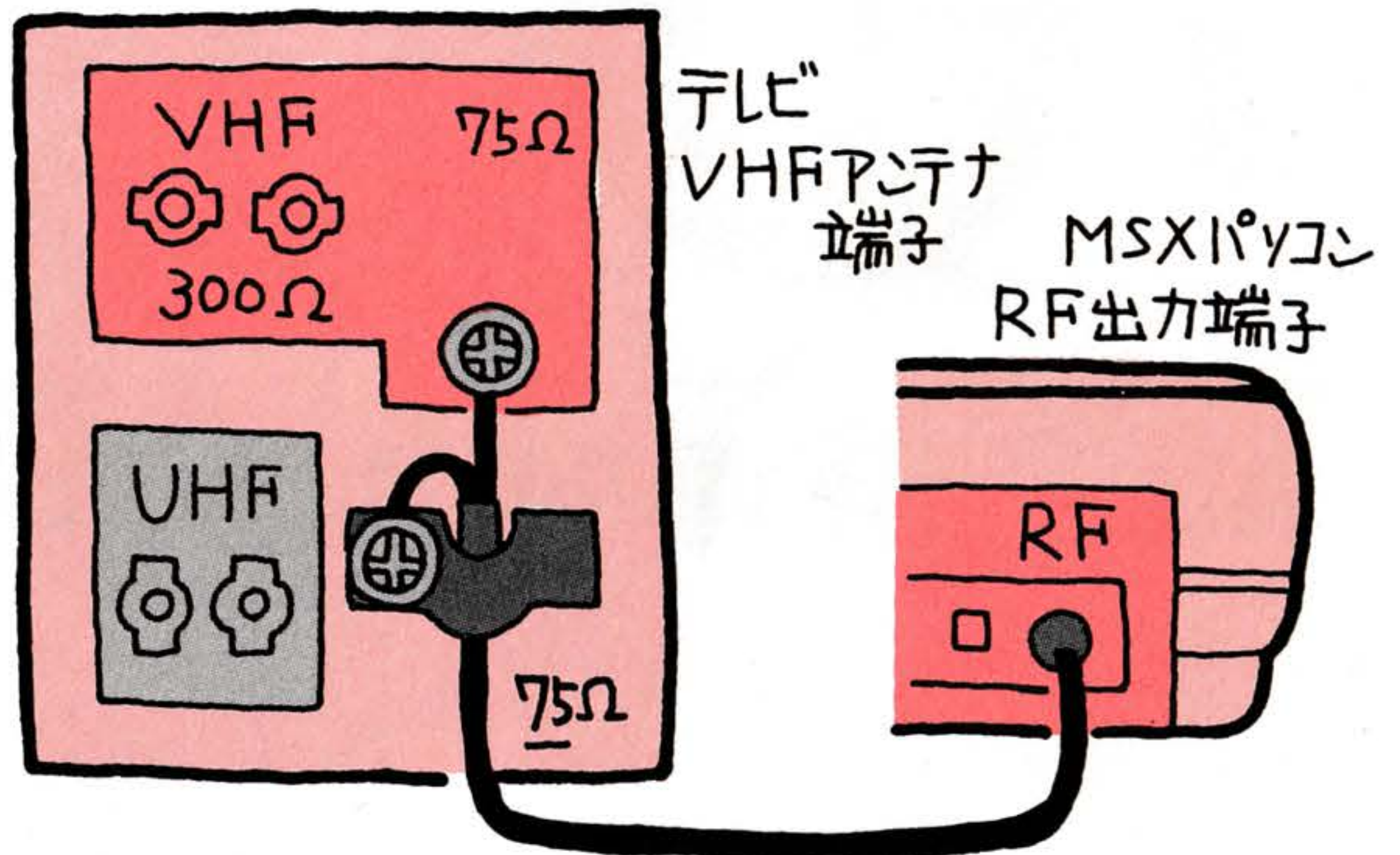
出力端子の内芯線、入力端子に、外芯線を接続する。

2. テレビの映像入力端子とVIDEO出力端子、テレビの音声入

力端子とAUDIO出力端子を接続する。

3. RGB端子を持つ

機種なら、1と同じ方法で専用テレビと接続する。

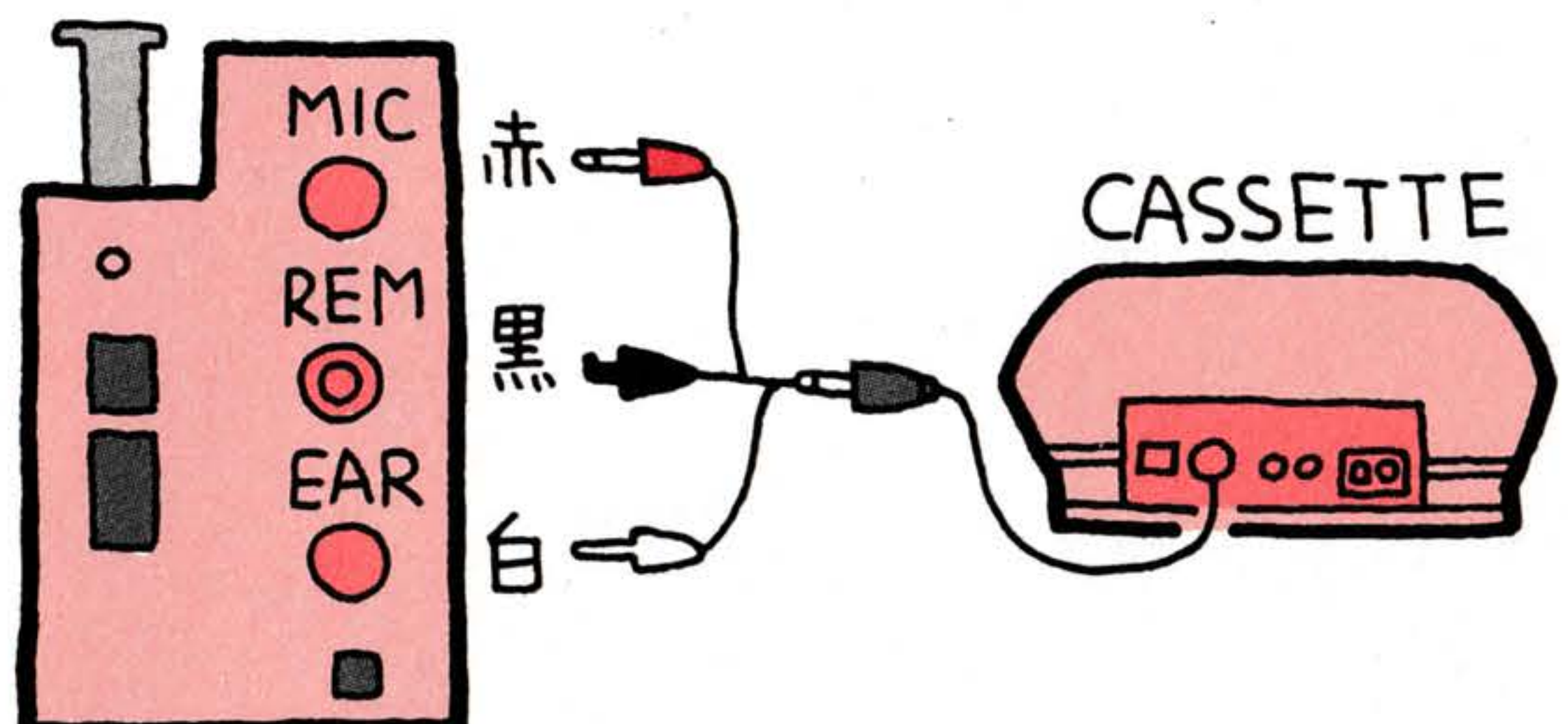


ジョイスティック

本体に2個の接続端子があるから、1台だけを使用するなら、1の端子と接続する。

テープレコーダ

本体の入力端子に、OUT（白）、IN（赤）、REMOTE（黒）を接続。





いよいよ スイッチONだ



しゅうへん き き さい ご ほん たい じゅん
まず周辺機器、最後に本体の順に

ほん たい しゅうへん き き せつぞく でん
本体と周辺機器の接続がすんだら、いよいよ電
げん
源スイッチをONして、パソコン操作自由自在だ。
といきたいところだが、あわててスイッチONし
てはいけないよ。

ほん たい しゅうへん き き て
本体でも周辺機器でも、手あたりしだいにスイ
ッチをONしていっては、故障のもとだよ。もの
には順序というものがある。スイッチONの順序
も、正しく守ってもらわなければならない。

スイッチONの順序は、まず周辺機器、それか
ら本体だ。テレビ（ディスプレイ）だけが接続さ
れているのなら、まずテレビのスイッチを入れて
から、本体・キーボードのスイッチを入れる。

プリンタやテープレコーダなど、ほかの周辺機
器が接続されているときは、周辺機器のどれを最
初にスイッチONしてもいいけれど、すべての周
辺機器をすましてから、本体のスイッチを入れる。

スイッチOFFの順序はその逆だ。まず最初に本
体・キーボードのスイッチを切る。それから周辺
機器のスイッチを切るという順序だ。

スイッチONにも
順序はあるのだ
守らないと
故障するかもよ



スイッチOFFは
逆の順序
まず本体の
電源を切ってから





はじめての画面表示は英語だ

本体・キーボードとテレビ（ディスプレイ）だけを接続したとして、実際に電源スイッチを入れてみよう。まずテレビの電源をONして、それから、本体のスイッチONの順序だよ。ここでは、SONYのHIT BITを参考にしよう。

あっ！とおどろくほどのことではないが、画面になにかが表示されたはず。文字だ。しかも、英語だよ。機種が違えば、多少は表示の内容も違うかもしれないけど、まあ、似たりよったりだから、多少の違いには目をつぶろう。

MSX system
version 1.0

Copyright 1983 by Microsoft

やがて、最初の画面から、次の画面に変わる。

SONY HIT BIT system

MENU

- ジュウシヨロク
- スケジュール
- メモ
- BASIC

カーソルキーでえらんでください RETURN

最初から

英語が表示されたって

こわがらなくてもいい

やさしい

英語なのだから



最初の画面。システム

ソフトの種類を示

している

カーソルの動かし方は

次のページで

よくおぼえてね



プログラムのメニュー

ーを選択する画面



これで操作はいつでもOKだ

前の画面に表示された文字を、よく見てみよう。

MENUと書いてあって、その下に、

- ジュウシヨロク
- スケジュール
- メモ
- BASIC

と書いてある。そして、その下を見てほしい。

カーソルキーでえらんでください

RETUR

N

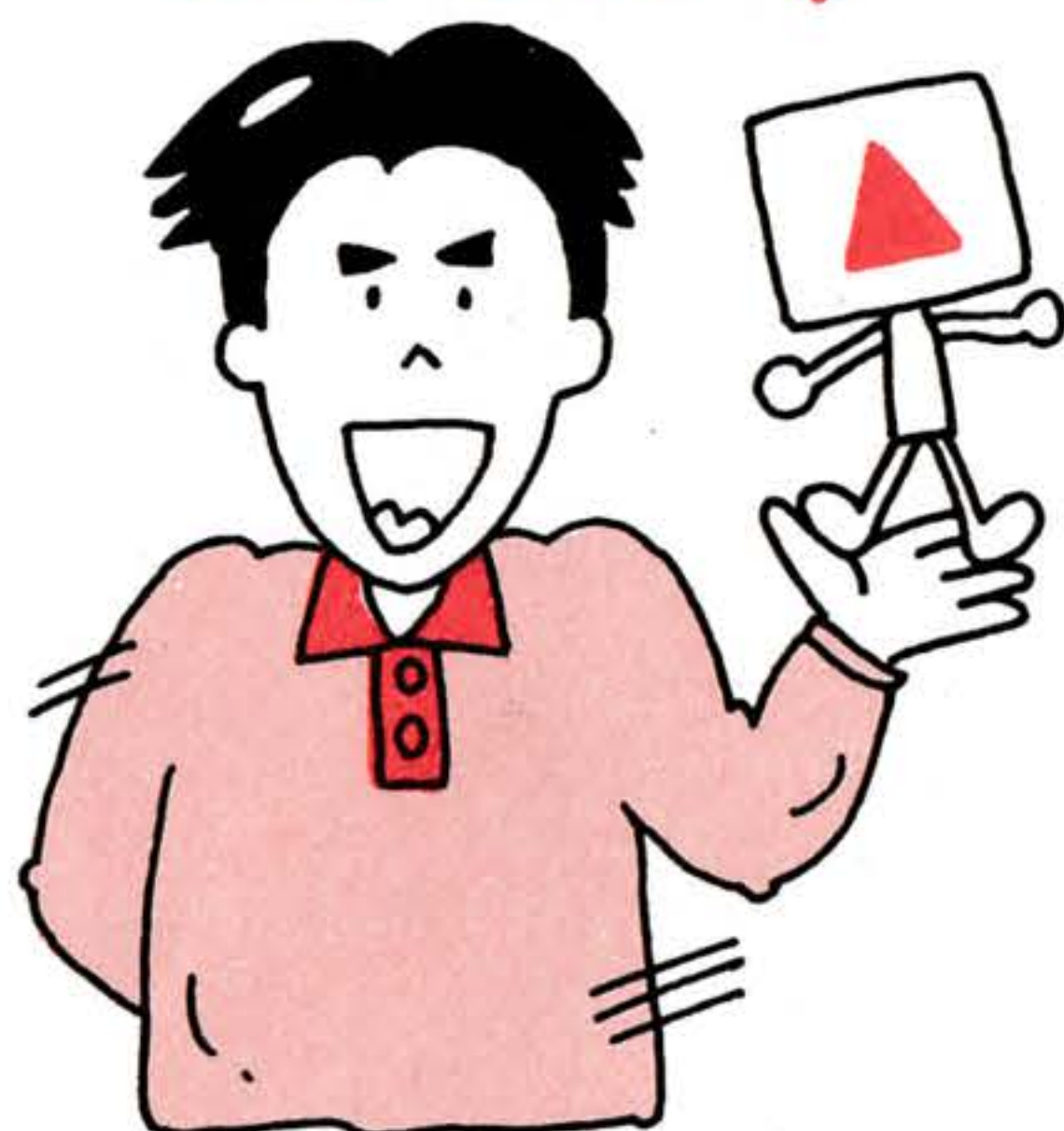
となっているから、そのとおりにしてみよう。カーソルキー▼を押して、画面にあるカーソルマーク(■)を、BASICと書かれた位置に移動させて、RETURNキーを押せばいい。

MSX BASIC version 1.0
Copyright 1983 by Microsoft
12431 Bytes free
OK

color auto goto list run

これで、パソコンを操作する準備は、すべて完了したというわけ。あとは、キーボードから、どんどん入力してみよう。

カーソルキーの使い方は
次のページにくわしくのっているよ

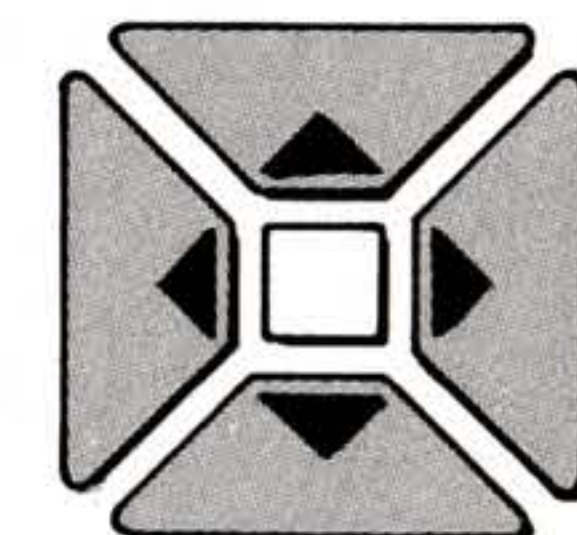
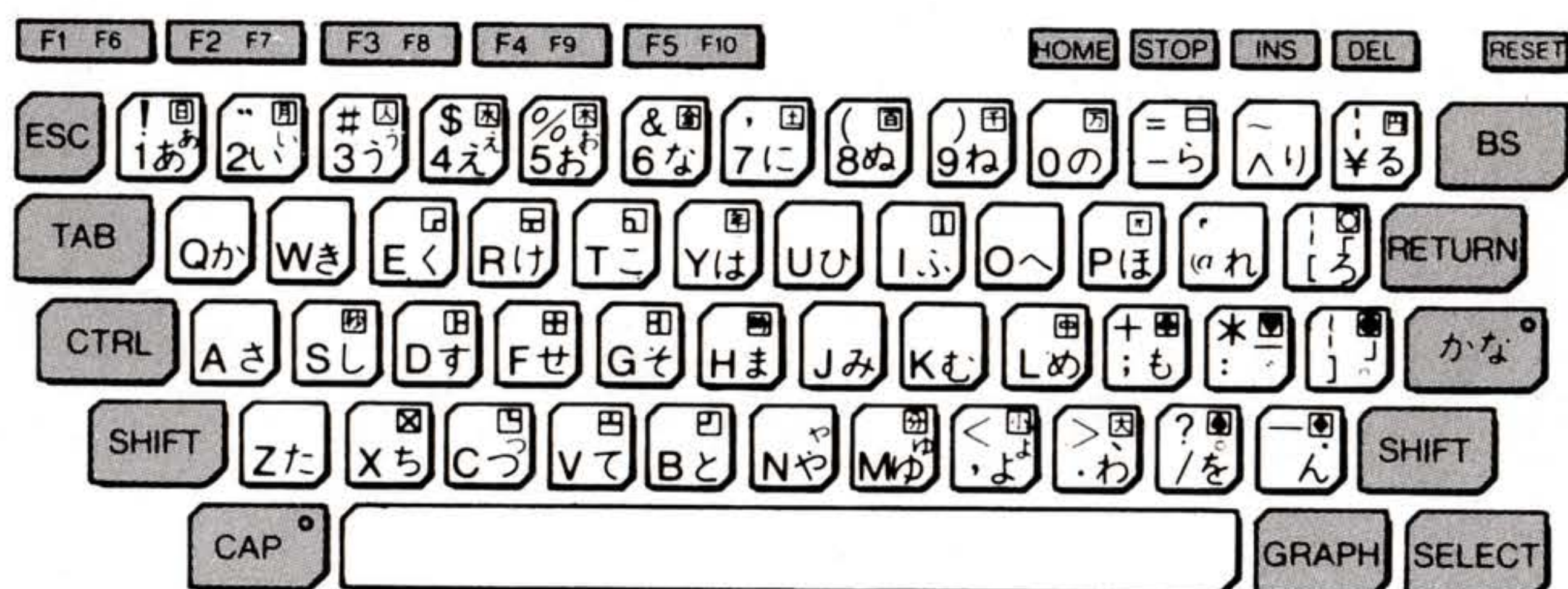


OKは
スタンバイOKだ
パソコン操作を
してもいいよ
というわけ



●カーソルキーの使い方

●キーボードとカーソルキー(SONY・HIT BITの場合)



カーソルキー

ボク、カーソルくん
てん のぼ
天まで昇って
いっちゃうよ～

ひだり い
左へ行って
がめん
画面のはじに
ぶつかっちゃうよ！

みぎ みぎ
右へ右へと
どんどん行って
ぶつかったら
じぎょう ぎょうとう
次行の行頭だ

お
押されるたびに
いっ ぽ いっ ぽ
一歩一歩
した
下におりて行くよ

カーソルとカーソルキー

がめん
画面にあるカーソル (■) の位置に、キーボードから入力した文
じ き ごう ひょう じ
字や記号が表示される。逆のいい方をすれば、入力する文字や記号
ひょう じ い ち しめ
が表示される位置を示すのが、カーソルだ。

カーソルは、画面の中で自由に動かすことができる。カーソルを
うご
動かすのが、カーソルキーだ。カーソルキーは矢印のついた▲▼▶
◀の4個のキーだ。これらのキーを押すと、カーソルは矢印の方向
い どう
に移動する。

11

キーボードから 文字や記号の入力



がめん 画面をきれいにしたあとで

がめん
画面にOKと出たら、^で
^{そう さ じゅん び}
操作準備OKだから、ど
んどんキー^{にゅうりょく}
入力してもいいのだが、^{が めん}
画面がよごれ
たままでは^{き い}
気に入らない。^{が めん}
画面のそうじをして、
きれいな^{が めん}
画面にしてから、レッツゴー！だ。^{が めん}
画面
にOKと^で
出たら、

C L Sとキーを押して

RETURNキーを押す

^{が めん}
画面がきれいになって、^{ひょう じ}
表示されているのは、
^{ひだりうえ}
左上にOKとカーソル(■)だけとなったはずだ。

さあ、いよいよ、^{も じ}
文字や^{き ごう}
記号のキーを押して、
^{も じ}
文字や^{き ごう}
記号を表示させてみよう。でも、ただやみ
くもにキーを押しても、^{おも}
思った^{も じ}
文字や^{き ごう}
記号は表示
されないよ。アルファベットや^{すう じ}
数字、かたかな、
ひらがな、^{き ごう}
記号にマークなど、^{も じ}
文字や^{き ごう}
記号の^{しゅるい}
種類
ごとに、キーの^お
押し^{かた}
方^{ちが}
が違^{ちが}
うのだ。

たとえば^あ
のキーを見^み
てみよう。



ひとつのキーにいろいろ^か
書いて
ある。これらから、^{にゅうりょく}
入力したり、
^{も じ}
文字を^{えら}
選ばなければならないのだ。

SHIFTキーを

お
押しながら

HOMEキーを

お
押しても

^{が めん}
画面は

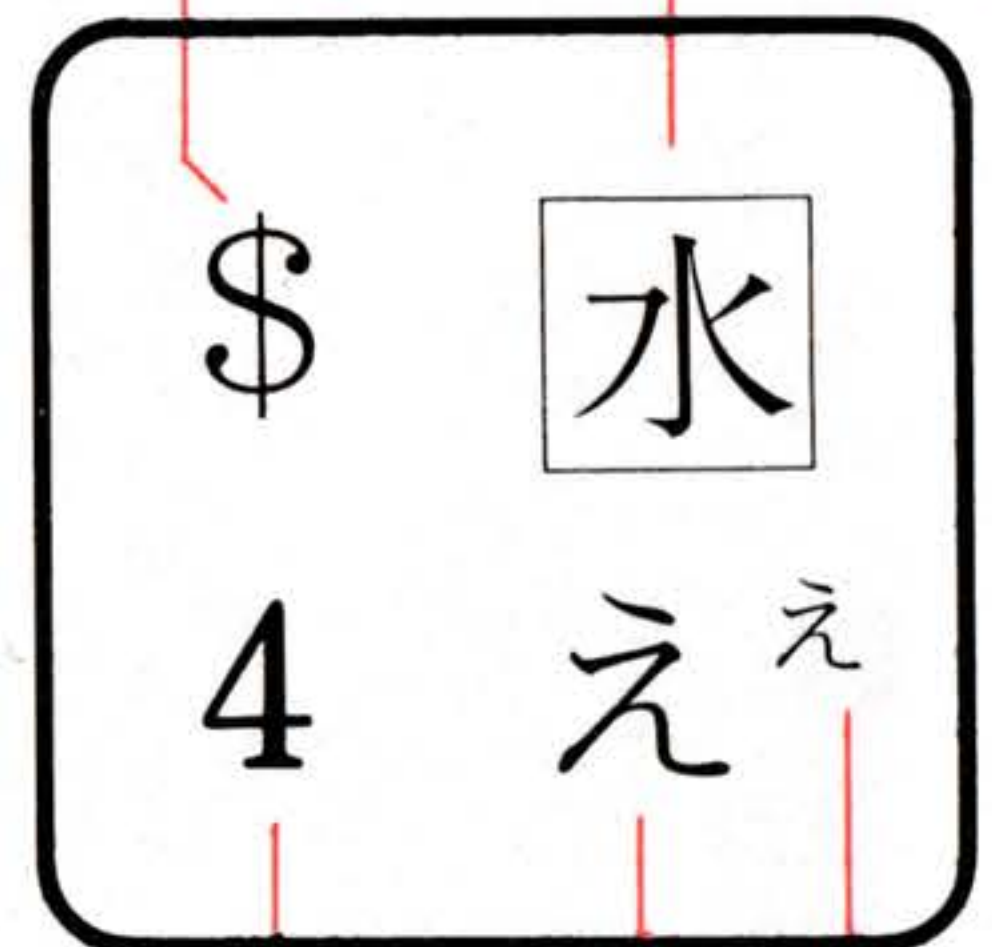
きれいになるよ



^{も じ}
文字キー

^{き ごう}
記号

^{も じ}
グラフィック文字



^{すう じ}
数字、^{くとう}
句読

^{てん}
点、^{ローマ}
ローマ

^じ
字の^{いち}
位置

^{こも じ}
小文字

ひらがな

●文字や記号の入力方法

文字や記号は、次のようにキーを打って入力する。

英小文字 → 英字

英大文字 → CAP 英字

ひらがな → かな 文字

ひらがな小文字 → かな SHIFT + 文字

カタカナ → かな CAP 文字

カタカナ小文字 → かな CAP SHIFT + 文字

数字 → 数字

記号 (キーの左下に表記) → 記号

記号 (キーの左上に表記) → SHIFT + 記号

グラフィック記号 → GRAPH + 記号

●キーに表記された位置での打ち分け

A = 数字、英小文字、記号 (一へ¥@ [; :] , . /) → キー

B = 記号 (! " # \$ % & , () = ~ ! ' { + * } < > ? -)

→ SHIFT + キー

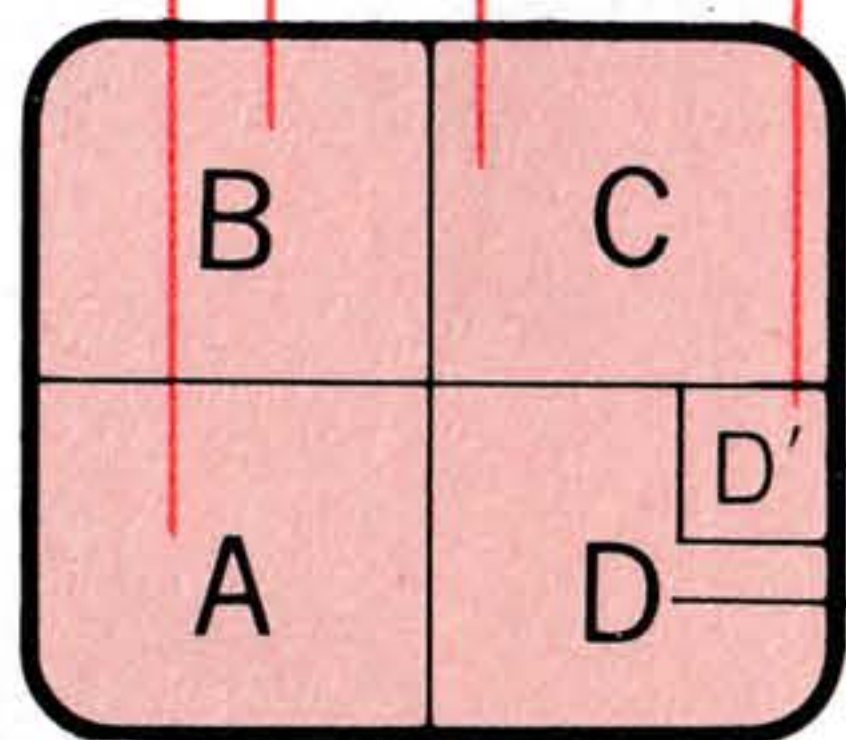
C = グラフィック記号 → GRAPH キー

D' = カタカナ小文字、ひらがな小文字、記号 (「一」、

。・) → かな SHIFT + キー (ひらがな)

または かな CAP SHIFT + キー

(ひらがなの小文字、記号)



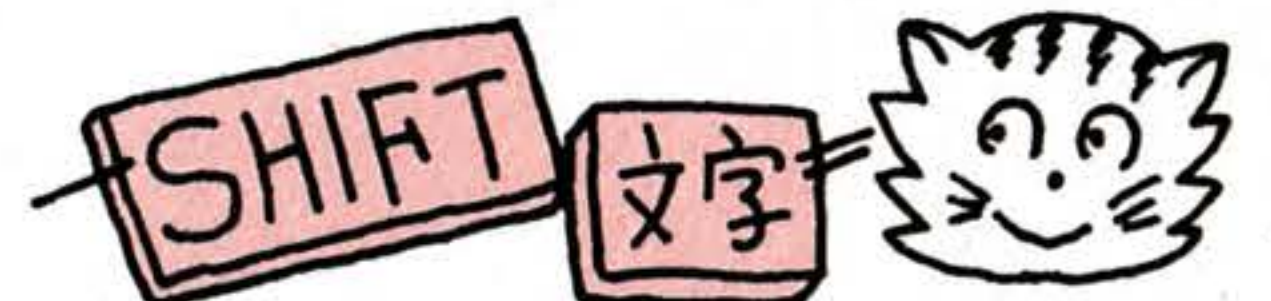
D = ひらがな → かな キー

カタカナ → かな CAP キー

もう1度 CAP を押せば
英小文字になるよ



SHIFT + 文字 は
SHIFT キーを
押しながら
文字 キーを押す



●ファンクションキーの使い方

ファンクションキーは、あるまとまった命令を
 入力したと同じ機能を持つキーだ。はじめに電源
 を入れたとき **F1** ~ **F10** までのファンクション
 キーは、キーを押すだけで次のような機能を果た
 すのだ。

●電源投入時のほたらき

F1 → COLOR

SHIFT + **F6** → COLOR 15, 4, 7 **RETURN**

F2 → AUTO

SHIFT + **F7** → CLOAD "

F3 → GOTO

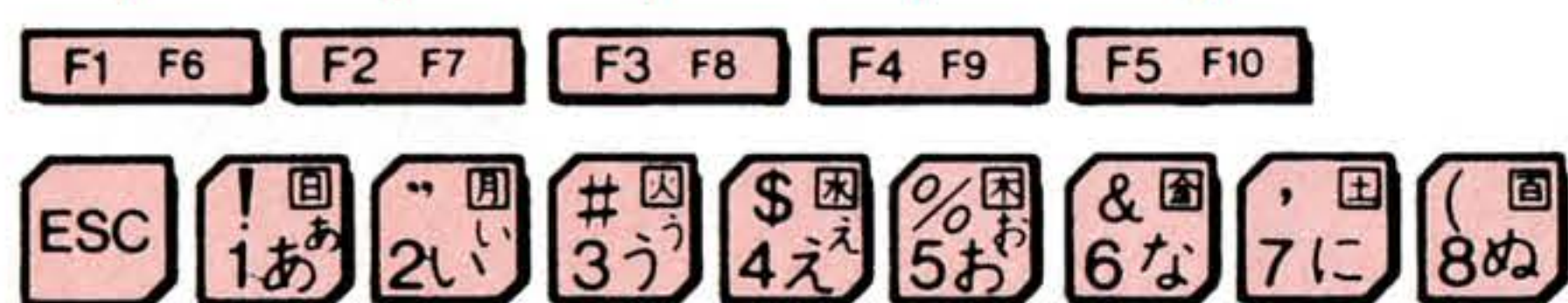
SHIFT + **F8** → CONT **RETURN**

F4 → LIST

SHIFT + **F9** → LIST **RETURN**

F5 → RUN **RETURN**

SHIFT + **F10** → CLS RUN **RETURN**



R U N RETURN

とするよりも

F5 キーをワンタッチ
 かんたんかんたん



LIST



よく出る命令を
 定義しておけば
 楽チンチン

●ファンクションキーに新しい機能を定義する

例 **F1** キーに PRINT を定義する

KEY 1, "PRINT" **RETURN**

定義したい命令語

定義したいファンクションキーの番号



012

プログラムは 保存しておこう



かえりよう
くり返し利用するために

カセットセーブ カセットロード ほぞん よこ
CSAVE&CLOAD ● 保存したり読み込んだり

せっかく入力したプログラムも、電源を切れば、
はかなく霧のように消えてしまうのがパソコンだ。
とっても苦心して作ったプログラムなのだから、
それを保存して、必要なときに、いつでも使える
ようにしなければ、なんのためのパソコンか、と
いいたくもなるだろう。

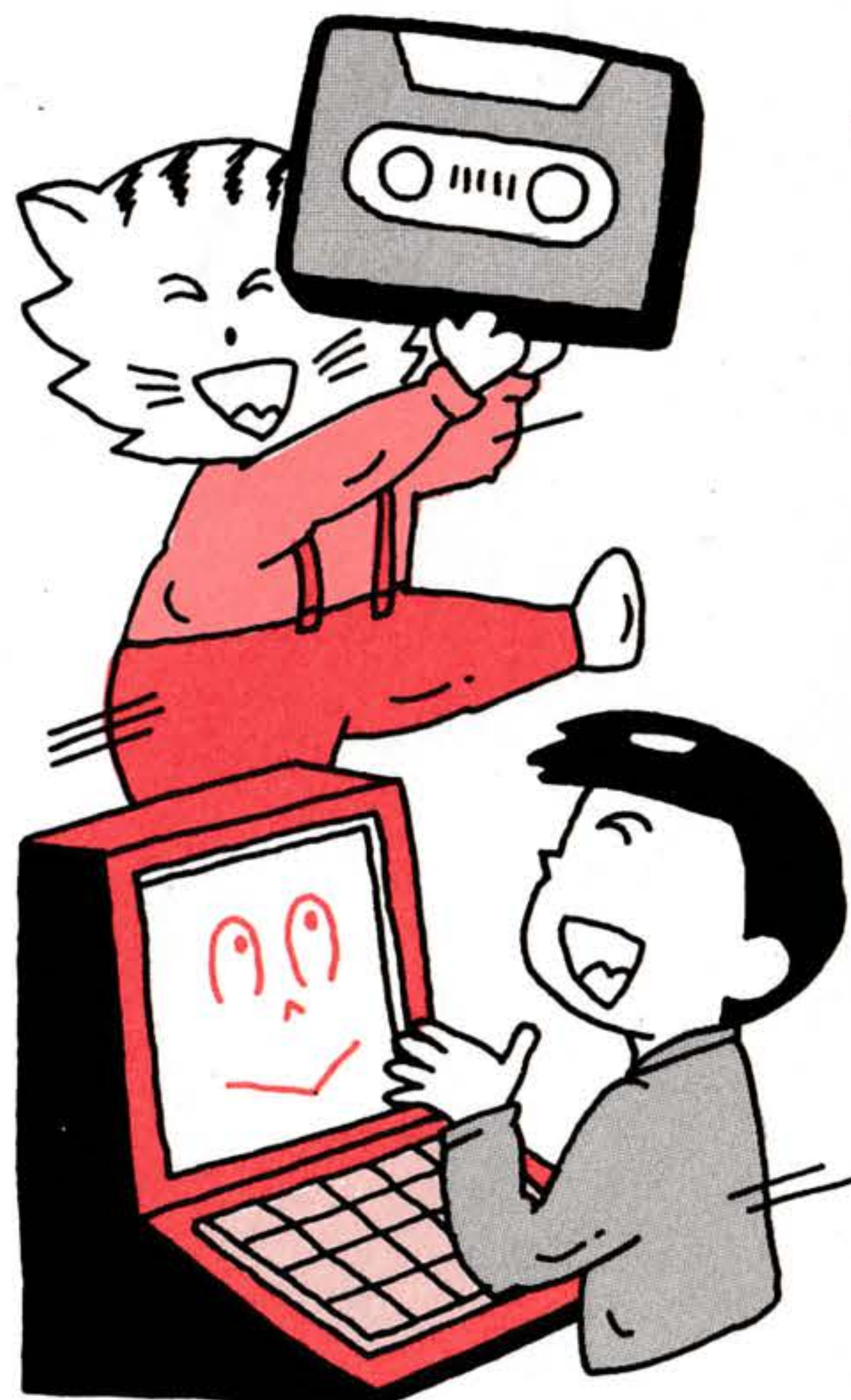
これを実現してくれるのが、CSAVEとSAVE
だ。

CSAVEは、プログラムをカセットテープに記
録して保存する。SAVEは、フロッピーディスク
に保存する命令だ。

そして、テープに記録したプログラムは、必要
なときにテープから呼び出して、パソコン本体に
読み込まなければならない。CLOADがその命令
だ。フロッピーディスクに記録したものを呼び出
すならLOAD命令だ。

とにかく、苦心して作ったプログラムは、CSA
VEかSAVEで、保存しておくことだ。

カセットテープに
記録するときの
テープはふつうの
音楽用テープで
いいんだよ



●CSAVE命令とCLOAD命令の使い方

●プログラムができたなら、名まえをつけよう

プログラムの名まえは6字以内だ。6字以上の名まえをつけてもいいが、MSXなら、6字までは読み取って、7字めからは、あってもなくても、ないのと同じだ。

たとえば、

マイコンOKコレクション

ここまで登録される

というプログラム名なら、「マイコンOK」までが、登録されたプログラム名だ。

1. キー入力 **CSAVE** “マイコンOK”
2. テープレコーダの録音スタート
3. **RETURN**

●FOUNDとOKが出たらLOAD OK

テープに保存したプログラムを読み出すなら、

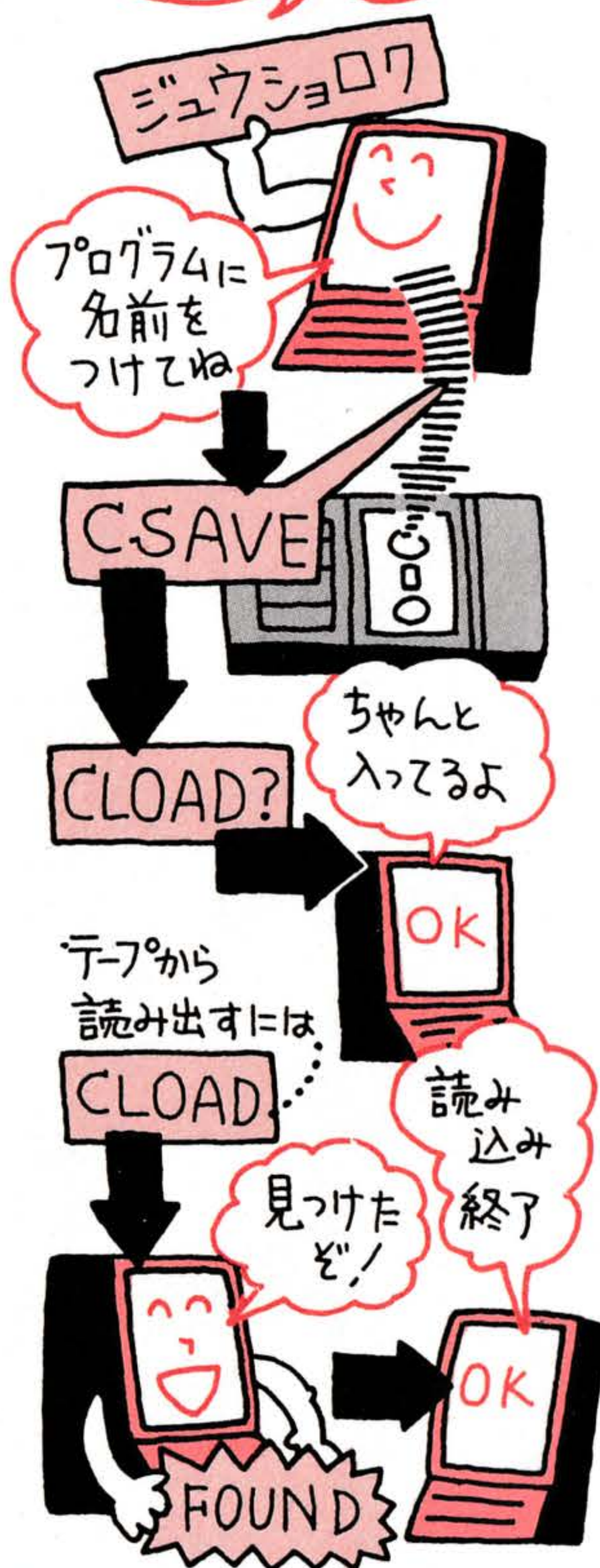
1. 電源を切って10秒ほど待って、もう1度電源を入れる。
2. キー入力 **CLOAD** “マイコンOK”
3. **RETURN**

読み込むプログラム名

テープの中に、読み込むプログラム名が見つかったとき、**FOUND**と表示される。

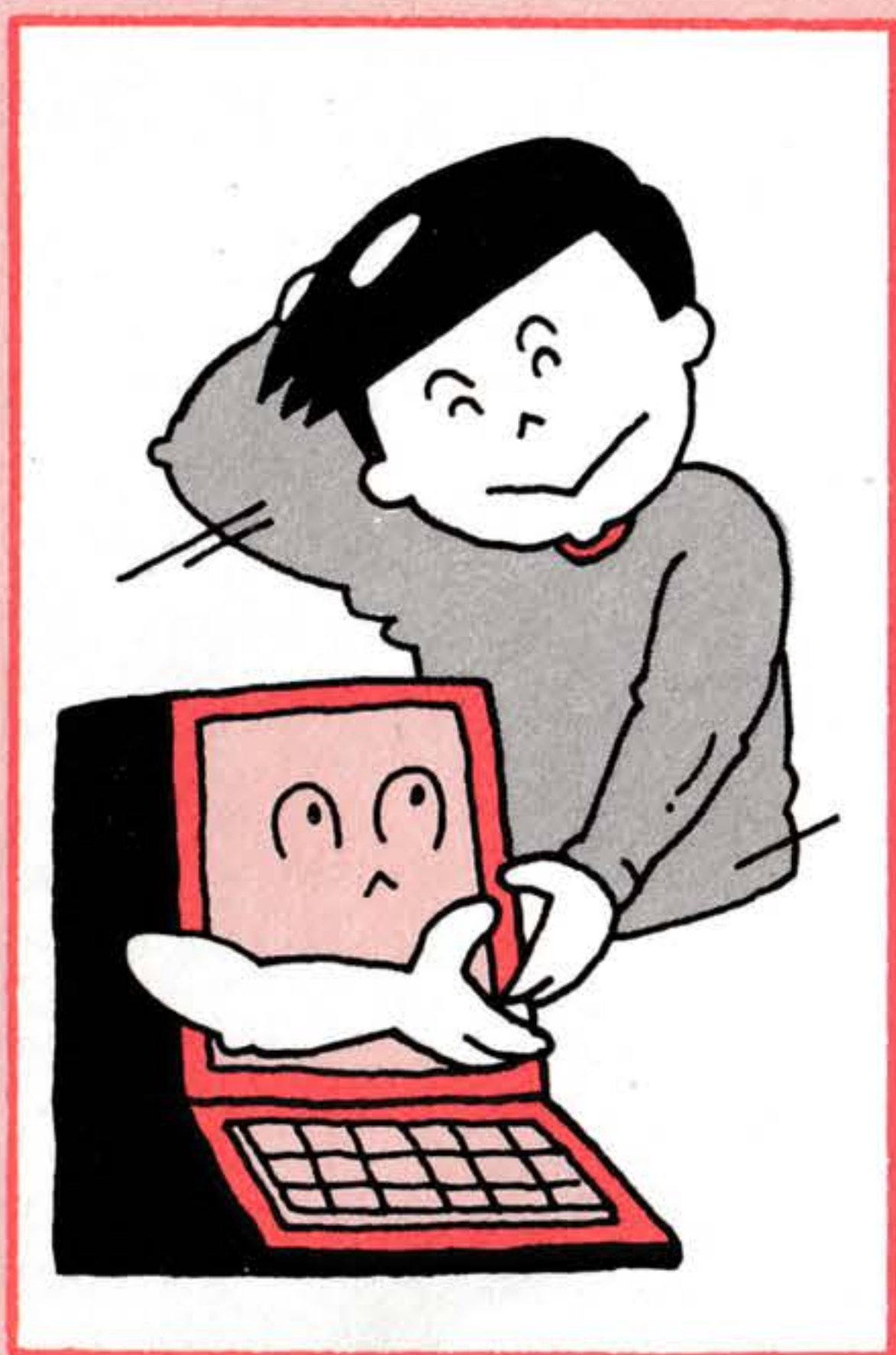
OKと出たら、プログラムの読み込みが終了したというサイン。

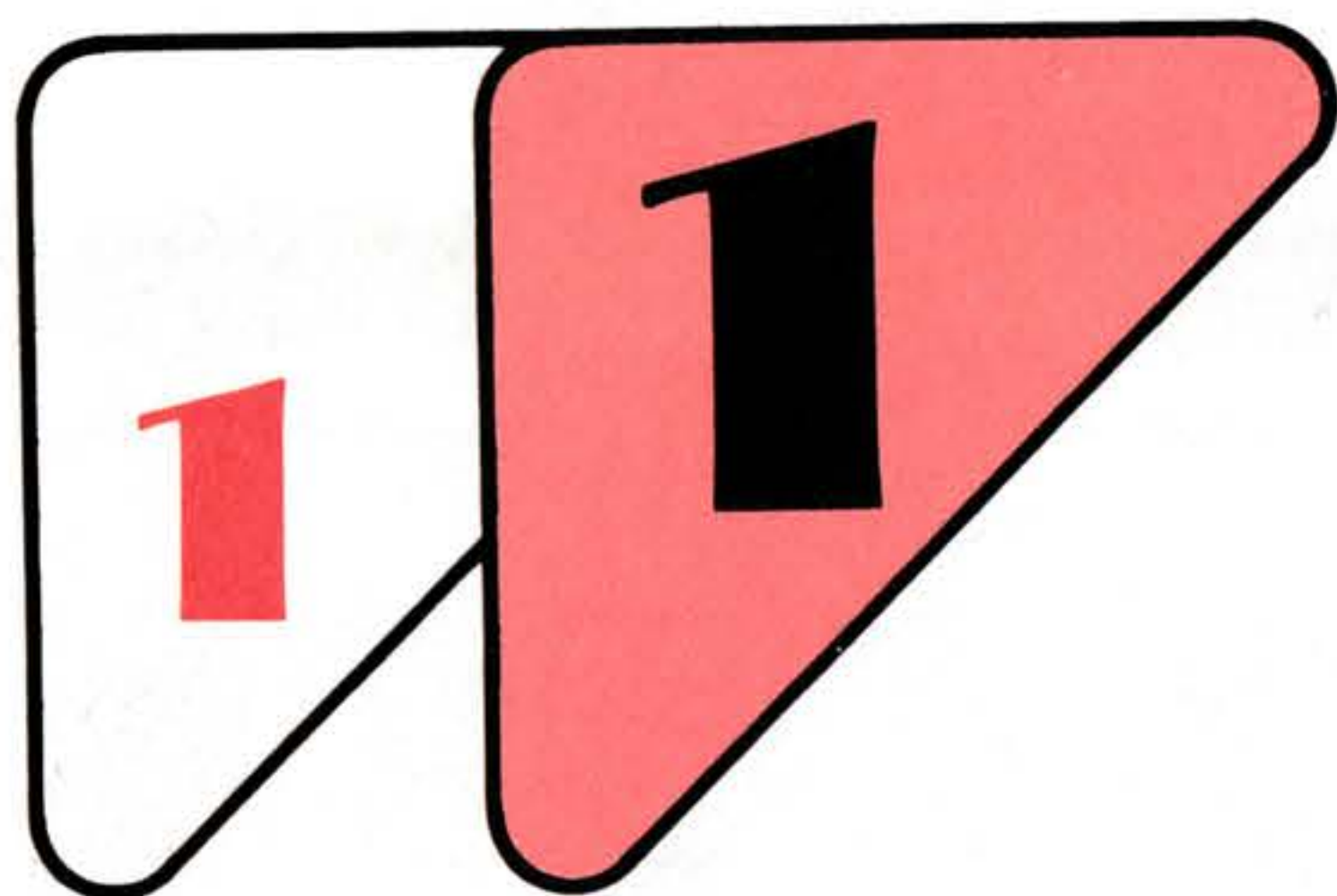
カセットセーブ
CSAVEが終わったら、電源を切る前に、プログラムがほんとうに入っているかを確認すること。
CLOAD? **RETURN** キー
でOKが出ればよし



パート 1

パソコン操作への 第一歩





パソコン計算機



けいさん き ごう

計算記号をおぼえよう

プリントめいれい PRINT命令その1 ● 計算して答えを画面に表示する

キーボードのことがだいたいわかったら、とにかくパソコンに何かしごとをさせてみようよ。

コンピュータは、もともと大型計算機として開発されたものだから、計算は得意中の得意。たとえば、地球のまわりをまわっている人工衛星の軌道や打ち上げるロケットの軌道も、むずかしい科学技術の計算も、みんなコンピュータの役目だ。

といっても、ここではそんなむずかしい計算をしようというんじゃない。きみたちの勉強や生活に役立つように、パソコンを計算機として使ってみようということなんだ。

この計算をしてくれるのがPRINT命令だ。

たとえば、 $3 + 5$ なら、

P R I N T **3** **+** **5**

とキー入力して **RETURN** キーを押す。

$10 - 4$ なら、

P R I N T **10** **-** **4**

とキー入力して、同じく **RETURN** キーを押すんだ。

RETURN キーを押し
わす
忘れると？

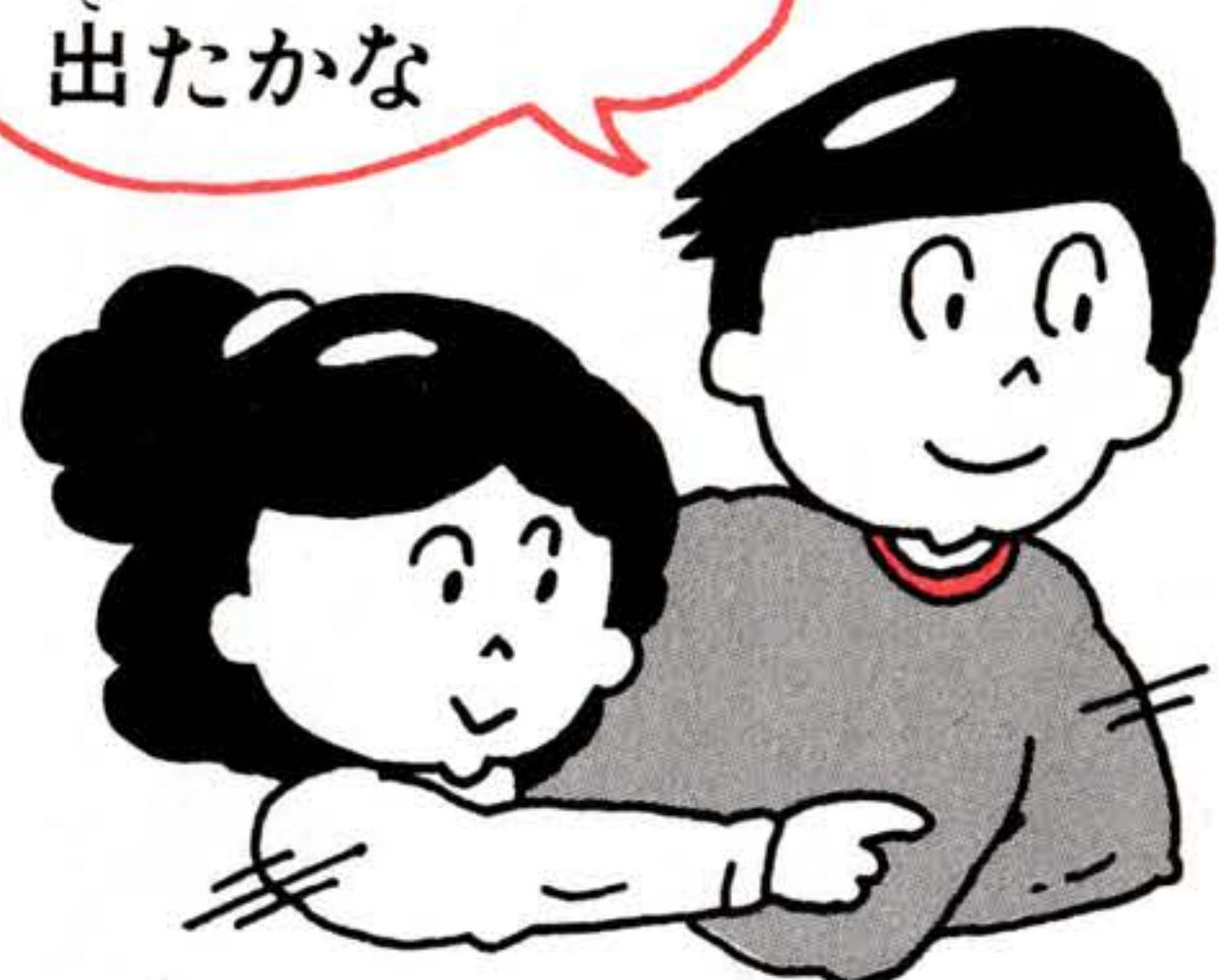


がめん
画面には文字が出て
もじ
いるけど、コンピュー
ターに命令が伝わら
ないよ。注意してね



●計算問題のいろいろ

さあ、画面には
どういうふうに出たかな



PRINT 3 + 5

8

OK

PRINT 10 - 4

6

OK



1行ごとにRET

URNキーを押さ
ないと、計算し
てくれないよ

RETURNキーを押すのは、
「命令を実行しなさい」と
いう意味なんだ

PRINT 8 / 2 + 5 * 6

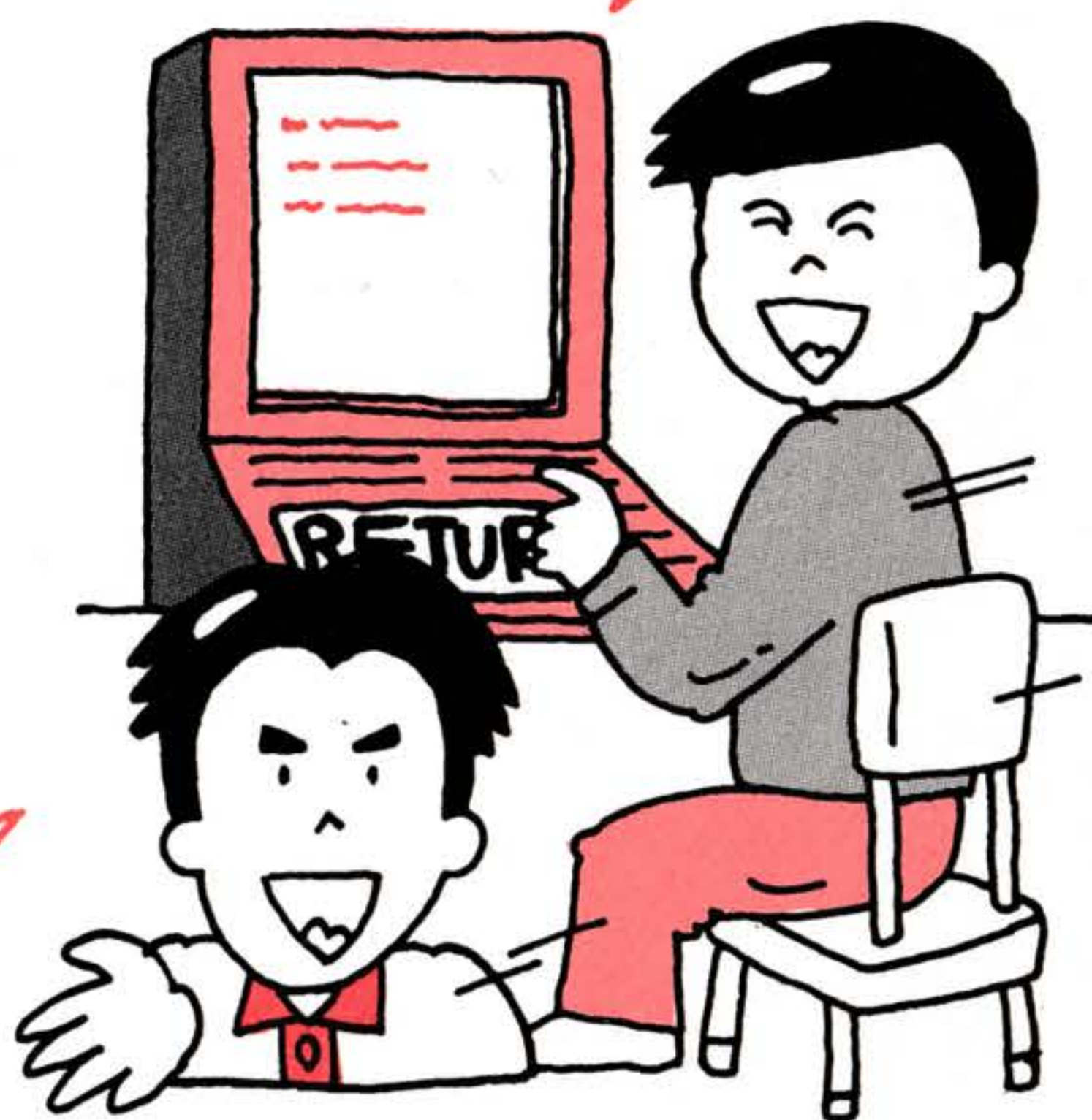
PRINT (30 + 5) / 7

PRINT (9 * (5 - 2)) / 3

PRINT 5 ^ 3 + 4 ^ 4

PRINT 9 ^ 4 / (27 * 3)

もう少しむずかしい計算をやらせて
みよう。たし算、ひき算よりかけ算、
わり算を先にするのはわかるね



コンピュータの計算記号

+.....タス (例) 5 + 3

-.....ヒク (例) 10 - 5

*.....カケル (例) 6 * 2

/.....ワル (例) 8 / 4

^.....べき乗 (例) 2 ^ 3 (2³)

()カッコ (例) 5 * (9 - 2)

() カッコがついて
いるときは、内がわの
カッコから計算する

(9 * (5 - 2)) / 3

さき
あと



ちやくせつ はな

直接話したりプログラムで話したり

はな

クリアスクリーン めい れい

が めん

CLS命令 ● 画面をきれいにそうじしてから

0章で、**C****L****S** とキー入力して **RETURN** キーを押すと、画面がきれいにそうじされたね。CLS命令は、それまで画面に表示されていたものをすべて消して、きれいな画面にするための命令なんだ。カーソルは左上に戻るよ。

では、まず画面をきれいにして……………。

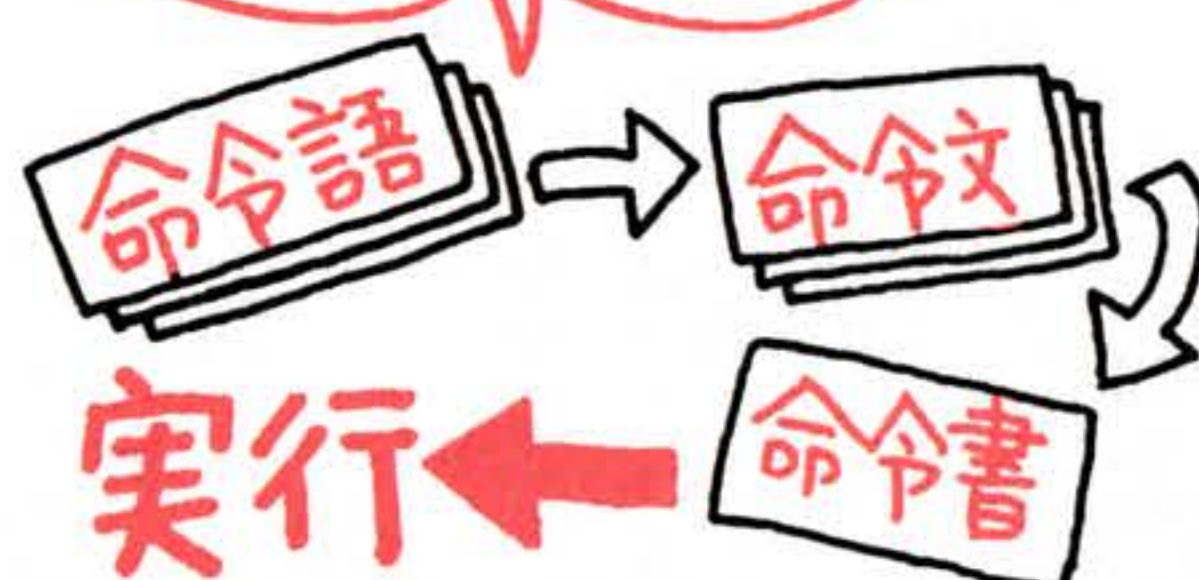
● パソコンに仕事をさせる2つの方法

パソコンに仕事をさせる方法は2通りある。ひとつは、今やった計算のように、直接キーボードから命令を入力して、**RETURN** キーを押し、その場で実行させる方法。もうひとつは、させたいしごとをプログラムにして、まずそれをパソコンに送り込む。そのあと **R****U****N** とキー入力し、**R****E****T****U****R****N** キーを押して実行させる方法だ。

キーボードから直接命令を入力して実行させる方法がダイレクトモード



プログラムを作ってそれを実行させる方法をプログラムモードというんだ



● プログラムってな～に？

プログラムは、パソコンにしごとさせるための命令書のようなものだ。この命令書はいくつもの命令文からできていて、しごとの順番にしたがって、1行ごとにきちんと番号がつけられて並んでいる。

まず、命令語を使って命令文を作る。次に、その命令文をどうやって並べて命令書を作るか。これがプログラミングなのだ。

プログラムを実行させる命令が、**R****U****N**命令である。

●プログラムを作ってみよう

さっきの計算をプログラム
にしてみよう

クリアスクリーンめい れい さいしよ い
CLS命令を最初に入れてお
くと、プログラムの実行の
ときに、自動的に画面をそ
うじしてくれるよ

```
10 CLS
20 PRINT 8 / 2 + 5 * 6
30 PRINT (30 + 5) / 7
40 PRINT (9 * (5 - 2)) / 3
50 PRINT 5 ^ 3 + 4 ^ 4
```



プログラムにつ
ける番号を行番
号というんだ

10番きざみにつ
けると便利だよ



34
5
9
381

さあ、答えを
出してみよう

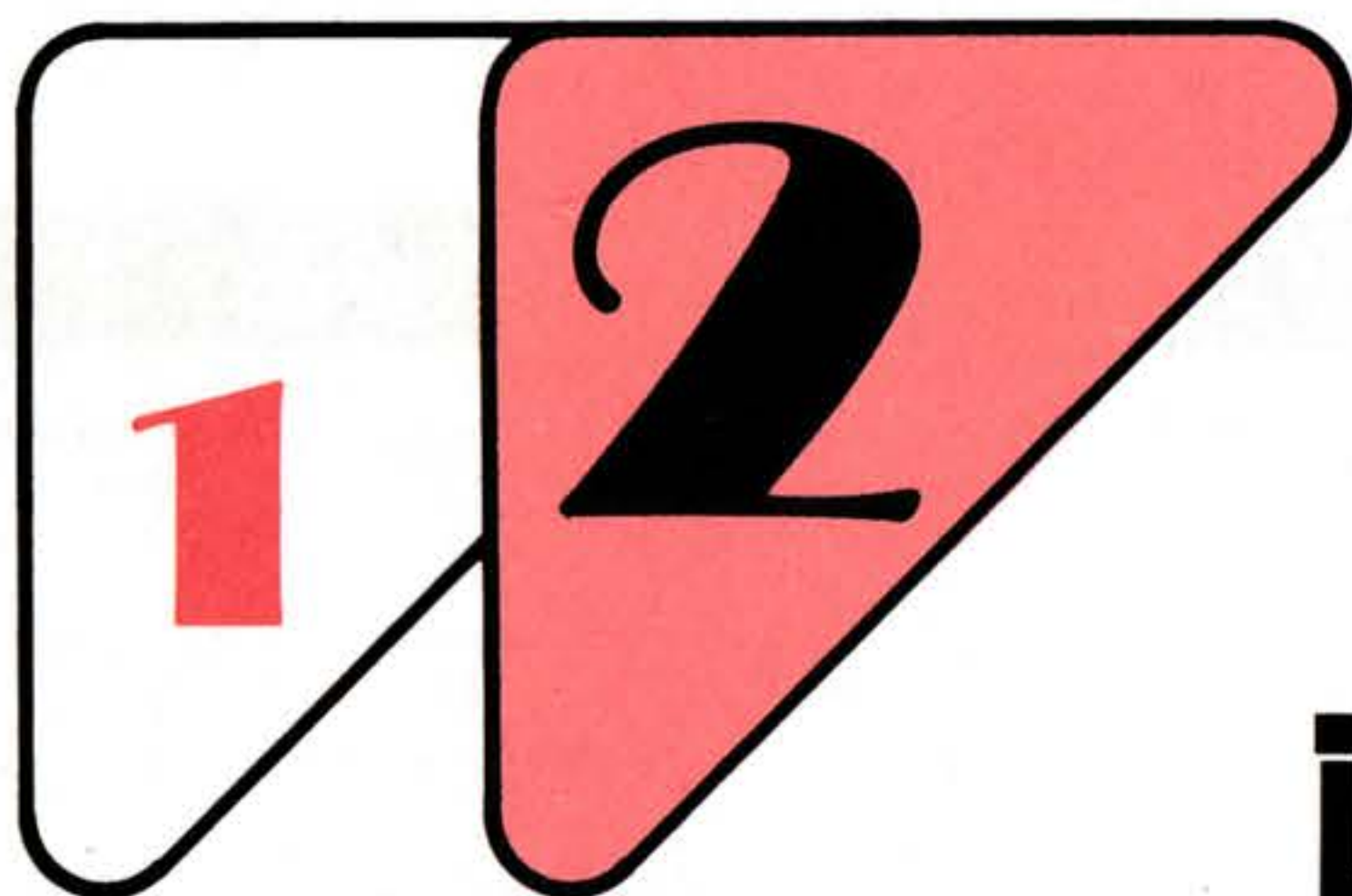
プログラムを実
行させるときは
RUN RET
URN キーだ



● RETURN キーの役わりは？

キーボードから直接、命令を入力するときの RETURN キーの役わ
りは、「命令を実行しなさい」という意味だ。

プログラムを入力するときは、番号のついている1行ごとに RET
URN キーを押さなければならないが、このときは「プログラムを記
憶しなさい」という意味になる。「記憶したプログラムを実行しなさい」というときは、 RUN RETURN キーだ。



画面に文字を書く



ひょうじ

表示するものを"

かこ

"で囲む

プリントめいれい PRINT命令その2 ● 文字やキャラクタを画面に表示する

プリントめいれい PRINT命令には、もうひとつの働きがある。

つぎ 次のプログラムを入力してみよう。

10 CLS

20 PRINT "3+5"

30 PRINT 3+5

さあ、キー入力してRUNさせてみよう。画面はどうなっただろう。

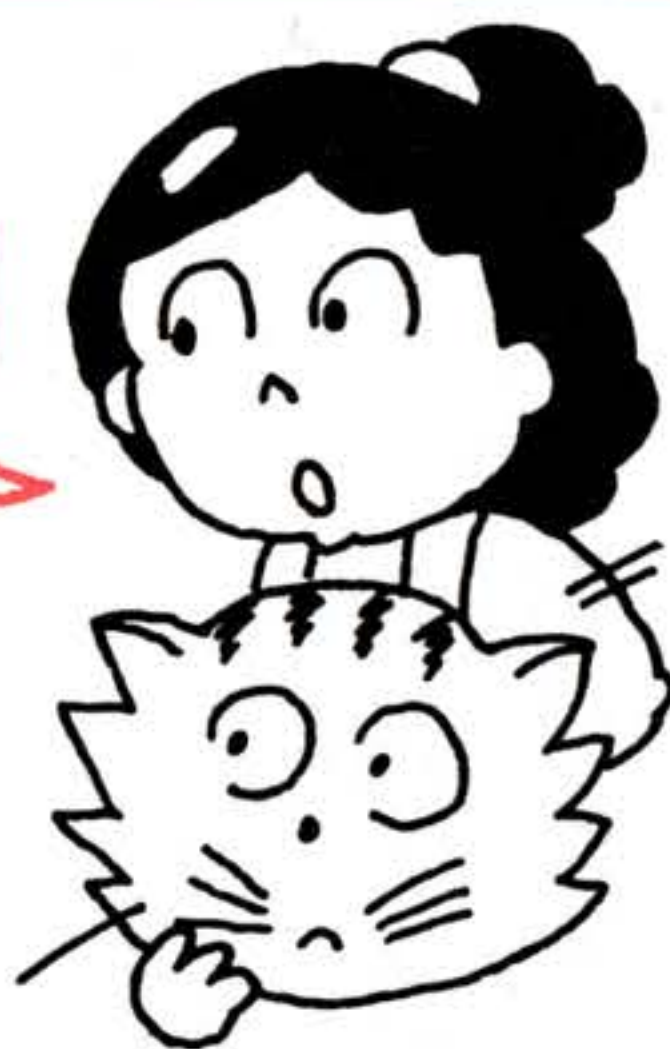
3+5

8

OK



おな プリントめい
同じPRINT命
れい
令なのに、どこ
ちが
が違うのかな？



アレッ!?

りょうほう

両方とも「3+5」

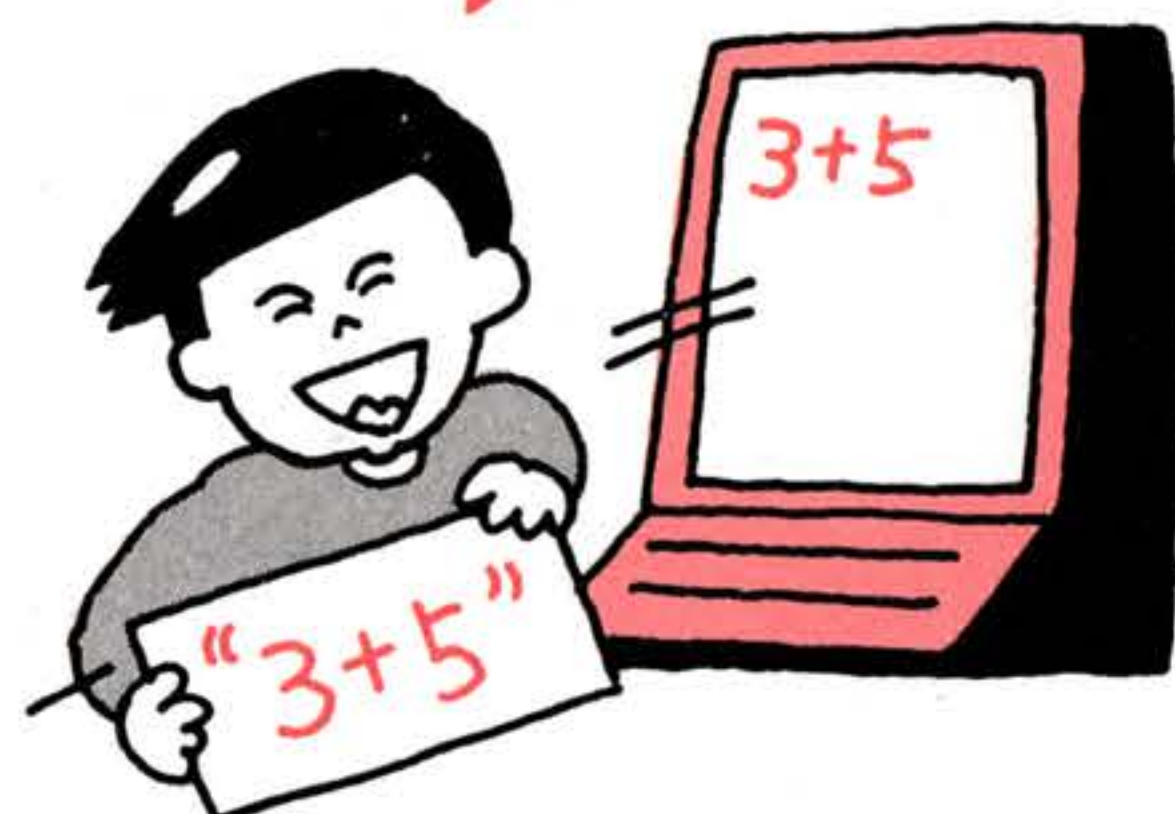
なのに、式と答えが
しき こた
で
出てきちゃったぞ



わかった！ " " が
ついているかいなか
ちが
で違うんだね

プリント PRINTのあとの文字や数字を " " (ダブルク
ォーテーション) でかこ 囲むと、「" " のなかを画面に
ひょうじ 表示しなさい」という意味になる。

だから20行のPRINT文では " " のなかをひょうじ 表示し
て、30行ではぎょう けいさん 計算をして画面にひょうじ 表示したんだ。



●アルファベットや模様を書こう

“ ” (ダブルクォーテーション) つきのPRINT命令がわかったところで、右のプログラムを入力してみよう。空白のところはスペースキーを押すんだ

```
10 CLS
20 PRINT"*****"
30 PRINT" * "
40 PRINT" * "
50 PRINT" * ABCDEFG * "
60 PRINT" * "
70 PRINT" * "
80 PRINT"*****"
```



キー入力の際に、“ ” を忘れないように注意しよう。もし忘れるとエラーが出てしまうぞ

エラーってなに?



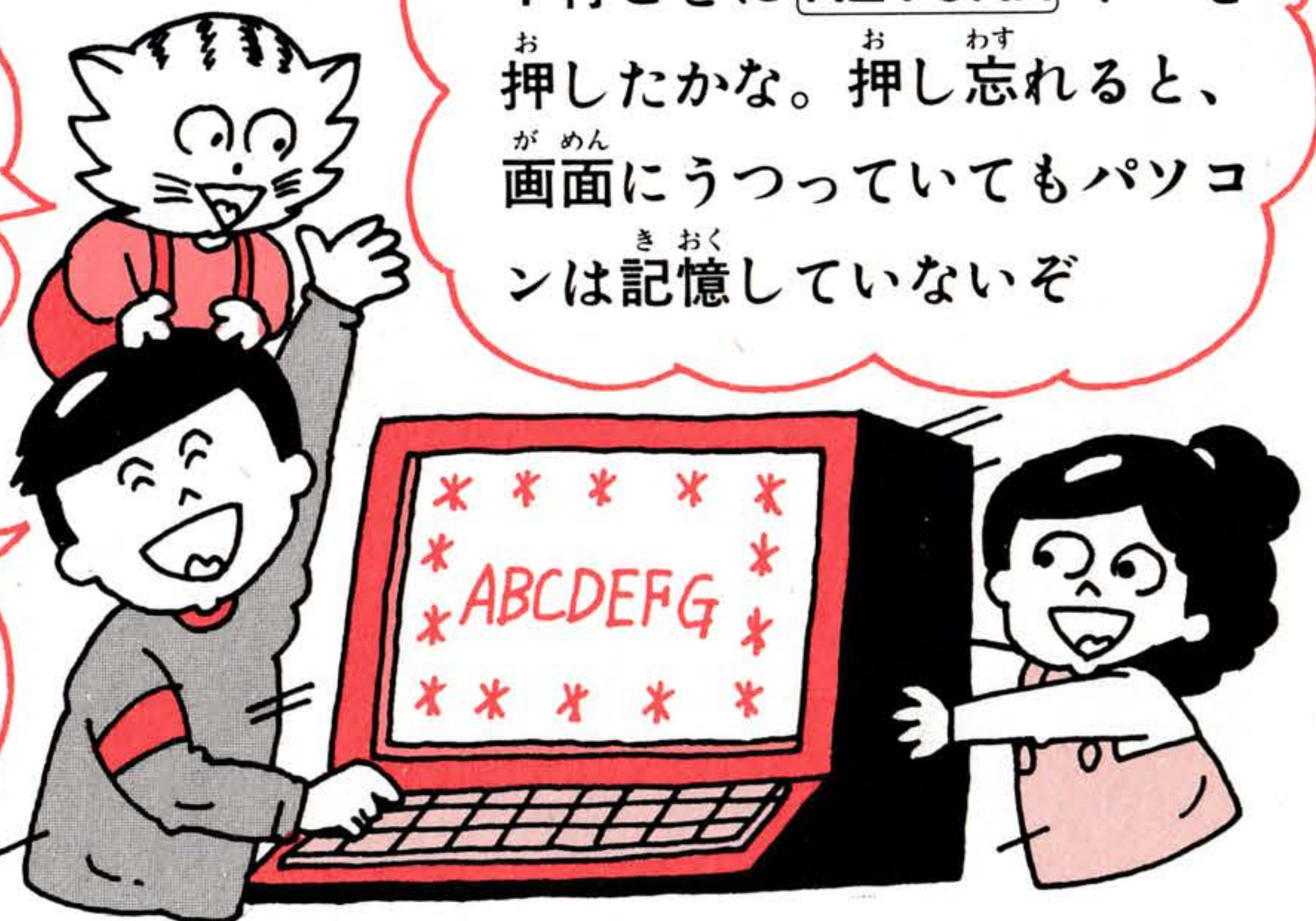
あとでくわしく説明するけど、プログラムのまちがいや入力ミスをパソコンが教えてくれる



OK! ぜんぶ入力できたよ。 RUN RETURN キーだね

1行ごとに RETURN キーを押したかな。押し忘れると、画面にうつっていてもパソコンは記憶していないぞ

あっ、出たぞ、出たぞ。*の箱の中に、アルファベットが書いてある。これがPRINT命令なんだね





まちがったときはどうしよう

リストめいれい ● プログラムを画面に表示する

さて、プログラムを実行しても、ちゃんと画面に表示されなかった人はいないかな。

プログラムにミスがあると、パソコンは途中で実行が止まってしまう。そのかわりに、「ピー」という音を出して、プログラムのどこにまちがいがあるかを教えてくれる。

おや？ こんな英語の文が出ているぞ。

Syntax error in 50

これは、「プログラムの書きかたがBASIC文法に合っていない。50行にそのまちがいがあります」と、教えているエラーメッセージだ。

では、プログラムを直さなければいけないね。
入力したプログラムを、もう一度画面に表示してくれる命令が、LIST命令だ。

LISTとキー入力して、RETURNキーを押してみよう。

```
10 CLS
20 PRINT"*****"
30 PRINT"*"
40 PRINT"*"
50 PRINT* ABCDEFG
60 PRINT"*"
70 PRINT"*"
80 PRINT"*****"
```

シンタックス エラー
Syntax error

が出たときは、キー入力のまちがいが多いんだよ



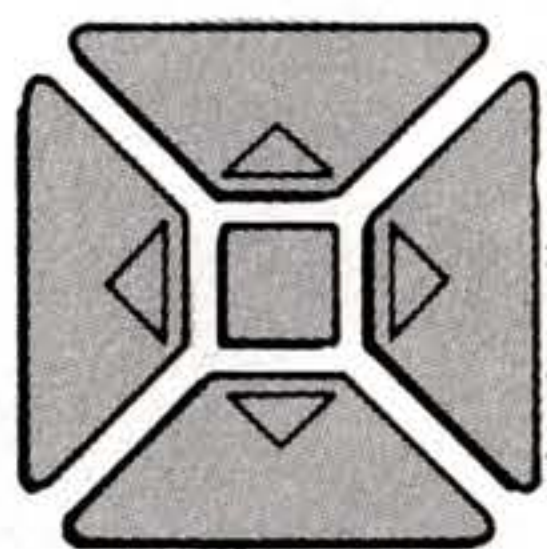
1文字1文字、正確にキーを押さなくてはいいけないね



アッ、ほんとうだ。50行に2つもまちがいがある。PRINTになっているし、"もぬけてるよ



●打ちまちがえた字を正しい字に直す

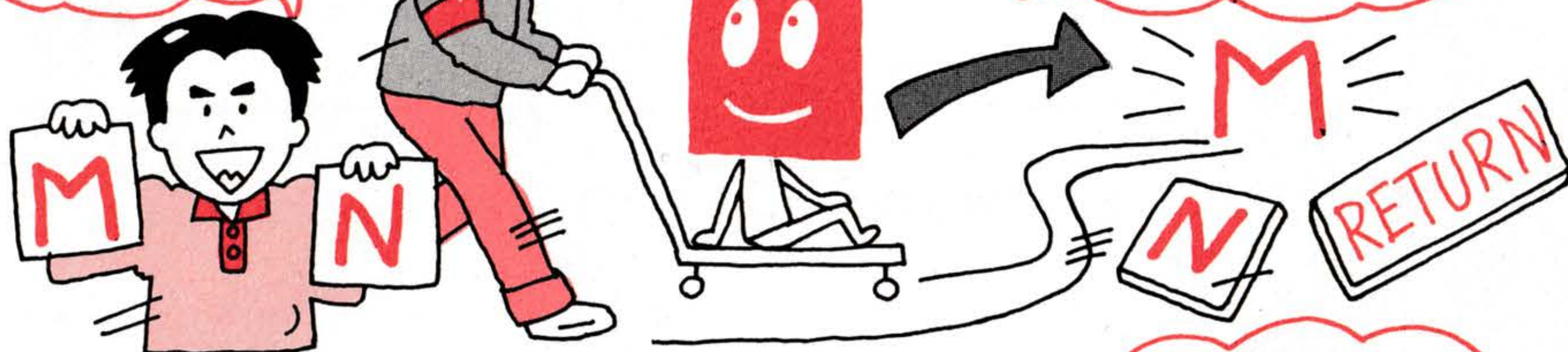


50 PRINT * ABCDEFG *

まず、PRINT
をPRINTに直
してやろう

カーソルをここまで
動かしてから

正しい文字Nを押
して、RETURN
キーを押すんだ



50 PRINT * ABCDEFG *

「がないと計算しちゃ
うけど、文字は計算で
きないから、エラーだ



よし、直ったぞ
あとは、PRINT
のあとに「を
入れてやらなく
ちゃ

LIST命令のいろいろな使い方

LIST 40.....40行のプログラムだけを表示させる

LIST 10-40.....10行から40行までを表示させる

LIST -40.....最初から40行までを表示させる

LIST 40-.....40行から最後までを表示させる

LISTプログラムをぜんぶ表示させる

●文字や記号を追加する

文字を追加するときに使うキーは、**INS**(インサート)キーだ。

50 PRINT * ABCDEFG *

追加したい字の右にカーソル移動だ

50 PRINT * ABCDEFG *

INS キーを押すと、カーソルが2分の1の大きさになり、

カーソルの位置をまちがえないで

50 PRINT " * ABCDEFG * "

" をキー入力すると " が追加される

50 PRINT " * ABCDEFG * "

もう一度 **INS** キーを押すと、カーソルが元にもどる

1 新しく文字を追加したい位置の右の字の上にカーソル ■ をかさねる。

2 **INS** (インサート) キーを押す。カーソル ■ が半分の大きさ ■ になる。

3 追加したい文字のキーをたたく。文字が追加されて、そのぶん前からあった文字が右側に動く。

4 追加し終わったら、もう一度 **INS** キーを押す。カーソルがもとの形 ■ に戻る。最後に、**RETURN** キーを押す。

終わったら、もう一度 **LIST** 命令でプログラムを画面に出してみよう。ちゃんと直っているかな？



●まちがえた文字を消す

まちがえた文字^{もじ}を消^けすときは、**[DEL]** (デリート) キー^{つか}を使う。

10 PRINT **T** "ABCDEF"

カーソル^け **■** を消^{もじ}す文字の
上^{うえ}にかさねる。

10 PRINT **■** "ABCDEF"

[DEL] キーを押^おす。カー
ソル位置^{い ち}の文字^{もじ}が消^きえる。

10 PRINT "A**B**CDEF"

↑
はじめのカーソル位置^{い ち}

右側^{みぎがわ}の字^じが1字分^{じ ぶん ぶん}左^{ひだり}に移^い
動^{どう}する。

10 PRINT "AF"

1回^{かい お}押すと1文字^{もじ}を消^けし
そのまま押^おし続けると押^おし
た回数^{かい すう}だけの文字数^{もじ すう}が消^{しょうきよ}去
される。BCDEをまとめて消^け
してみよう。

●画面表示中のリストを途中でストップするには？

短い^{みじか}プログラムであれば、一度^{いち ど}にぜんぶのプログラムが画面^{がめん ひょう}に表
示^じされるが、長い^{なが}プログラムになると、上^{うへ}の1行^{ぎょう}から順番^{じゅんばん}に消^きえて、
下^{した}から続^{つづ}きのプログラムがどんどん表示^{ひょう じ}されてくる。

この途中^{とちゅう}で画面^{がめん}を停止^{ていし}させたいときは、**[STOP]** (ストップ) キー
を押^おして表示^{ひょう じ}の実行^{じっこう}を中断^{ちゅうだん}させる。もう一度^{いち ど} **[STOP]** キーを押^おせば、
ふたたび表示^{ひょう じ}が続^{つづ}けられる。

また、途中^{とちゅう}でLIST命令^{めいれい}の実行^{じっこう}を終わ^おらせたいときには、**[CTRL]** キー
と**[STOP]** キーを押^おす。



好きな位置に表示する



がめん
画面には29×24のマス目がある

ローケート ● 画面の縦の位置と横の位置を指定する

さっきPRINT文で実行させた模様とアルファベ
ットは、画面のどこに表示されたかな？ いちば
ん上の左はしから表示されたね。

では、画面のまん中とか下の方とか、自分の好
きなところに表示させるにはどうしたらいいか。

画面に表示する位置を決めるのが、LOCATE命
令だ。

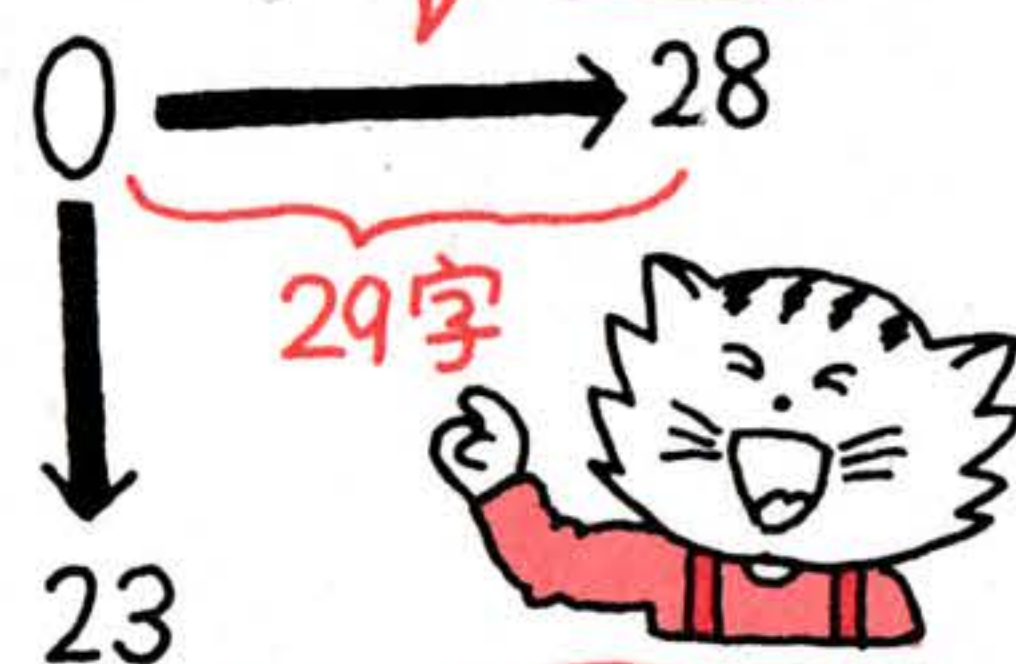
文字を書く画面は、横に29字、縦に24行の字が
表示されるようになっている。位置を決めるため
に、横をX座標として0から28まで、縦をY座標
として0から23までの番号が決められている。

この座標の位置をLOCATE命令で指定すると、
その位置からはじめの1文字めが表示されるんだ。

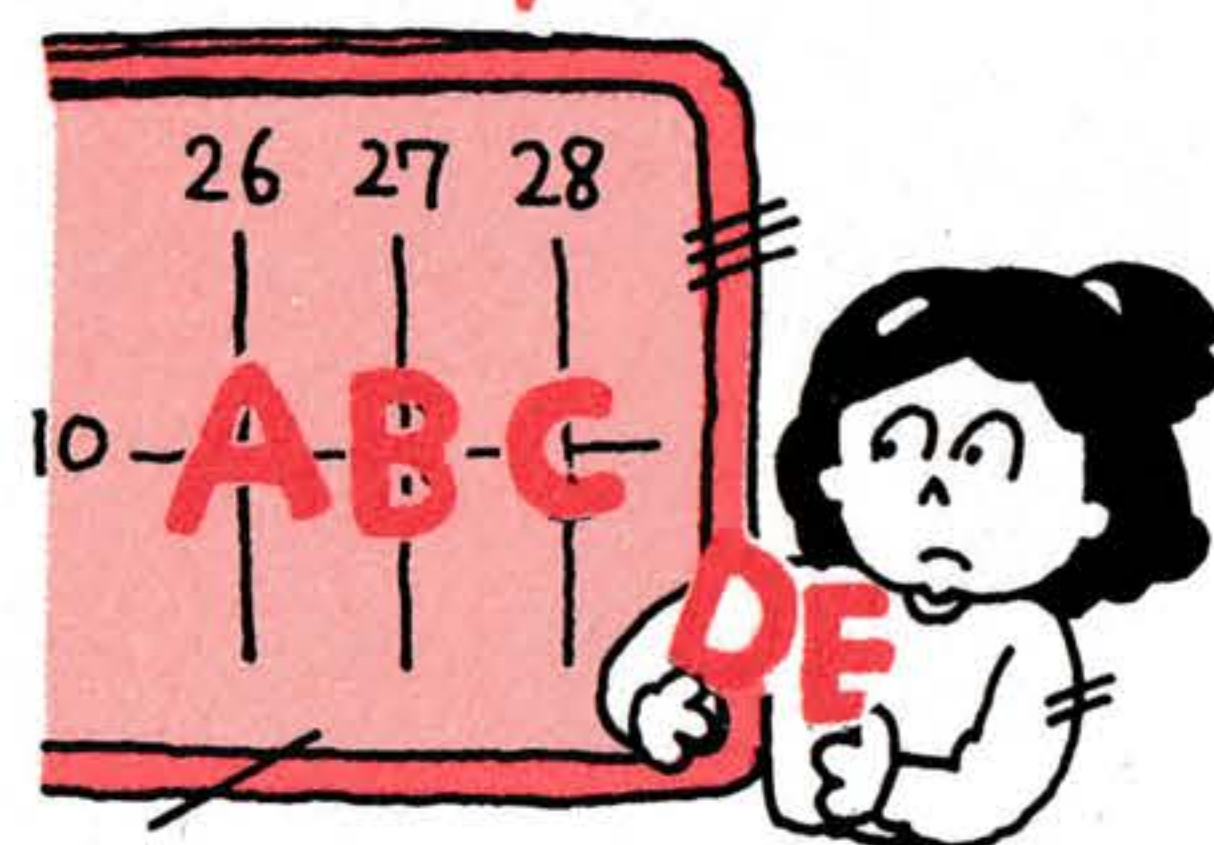
次の式のX、Yにいろいろな数値を入れてため
てごらん。ただし、Xは0から28、Yは0から
23までと決められていることに注意してね。

LOCATE X座標, Y座標

指定した位置から1文
字めが始まるんだから、
文字数にも注意しよう



ABCDEを続けて
表示させたいのに
LOCATE 26,
10なんて指定すると、
DEだけ次の行の頭
から表示されてしま
う



●ひとつのマス目に1字表示

```
10 CLS
20 LOCATE 10, 5
30 PRINT "ABC"
40 END
```

Aの字がどこから表示されるか
見てみよう



	0	1	2	3	4	5	6	7	8	9	10	28
0													
1													
2													
3													
4													
5													
6													
7													
8													
9													
.....													
23													

文字や記号を
表示させる画面
がテキストモー
ドだ。画面の縦、
横の番号は0か
ら始まることに
注意しよう。

次のプログラムを実行してみよう。

```
10 CLS
20 LOCATE 2,3:PRINT "MSX"
30 LOCATE 10,10:PRINT "♥♥♥♥♥♥♥♥"
40 LOCATE 10,11:PRINT "♥"
50 LOCATE 10,12:PRINT "♥"
60 LOCATE 10,13:PRINT "♥"
80 LOCATE 10,14:PRINT "♥♥♥♥♥♥♥♥"
90 LOCATE 12,12:PRINT "MSX"
```



上のプログラムの意味はわかったかな？ 20行で横3マスめ、縦4マスめのところから「MSX」を表示する。そのあと、♥で箱を書き、その中にMSX(90行)を表示する。こうして、グラフィックキャラクタを使えば、かんたんな模様なども表示させることができるよ。

●グラフィックキャラクタを表示するプログラム

グラフィックキャラクタを使って、トランプの「スペードの3」と「戦車」を表示させてみた。

「スペードの3」を表示する

```
10 CLS
20 LOCATE 3,3:PRINT"┌───┐"
30 LOCATE 3,4:PRINT"| ♠ |"
40 LOCATE 3,5:PRINT"| ♠ |"
50 LOCATE 3,6:PRINT"| ♠ |"
60 LOCATE 3,7:PRINT"└───┘"
70 END
```

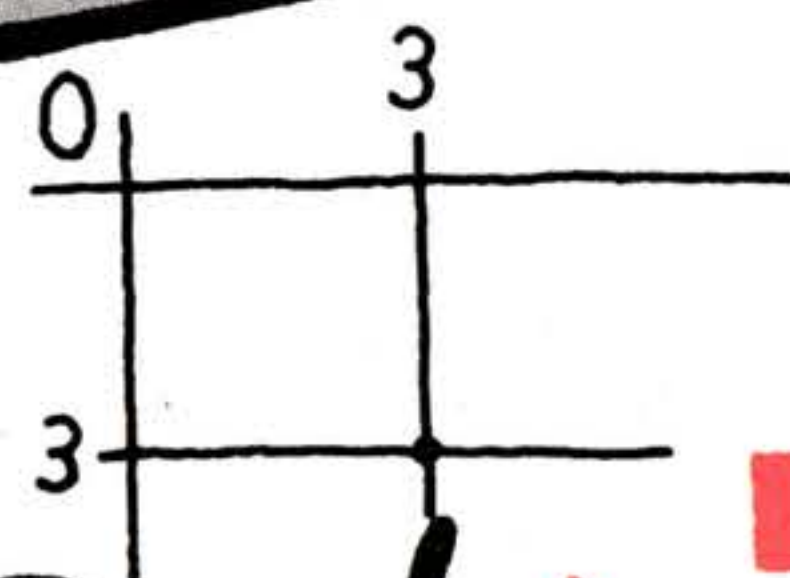
「戦車」を表示する

```
10 CLS
20 LOCATE 10,10
30 PRINT"---[#]"
40 LOCATE 10,11
50 PRINT"  [#]"
60 LOCATE 10,12
70 PRINT"<*****>"
```

ローケートめいれい プリ
LOCATE命令とPRI
ントめいれい く あ
NT命令を組み合わ
せれば、いろいろな
もよう ひょうじ
模様を表示させるこ
とができるよ

LOCATE

PRINT

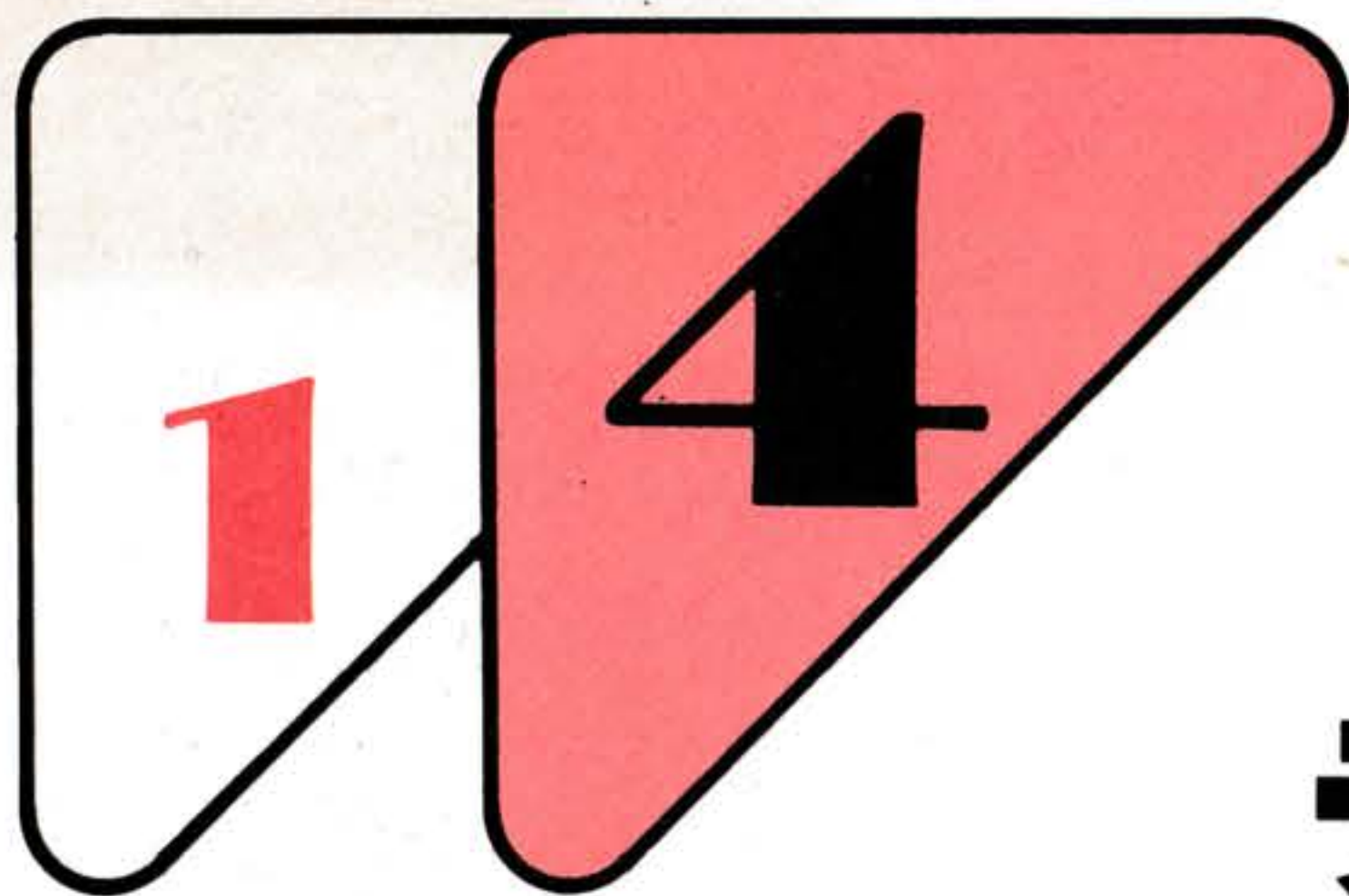


位置
を指定

画面
に表示

めいれい めいれい やく
命令と命令をつなぐ：（コロン）の役わり

左のプログラムは、ひとつの行に、LOCATEとPRINTがいっしょに並んでいて、それぞれの命令が：（コロン）で区切られている。：を使うと、ひとつの行の中にいくつもの命令をつなげて書くことができる。これをマルチステートメントというが、あまり使いすぎると見にくくなるので注意しよう。



データを入れる箱



かず い はこ すう ち へんすう
数を入れる箱 "数値変数"

= (イコール記号) は「等しい」ではなく「代入する」の意味

「変数」は、命令とは違うけれど、プログラムを組むうえで、たいせつな考え方のひとつだ。

次の計算問題をやってみよう。

Aの値は10、Bの値は20、Cの値は30です。
D = A + B + CのDの値はいくつでしょう。

かんたんな問題だね。AとBとCの値をたした答えDは60だ。

このA、B、C、Dが、コンピュータのプログラムの中で「変数」とよばれるものなんだ。

この計算問題をプログラムに直してみよう。

```
10 CLS
20 A=10:B=20:C=30
30 D=A+B+C
40 PRINT D
```

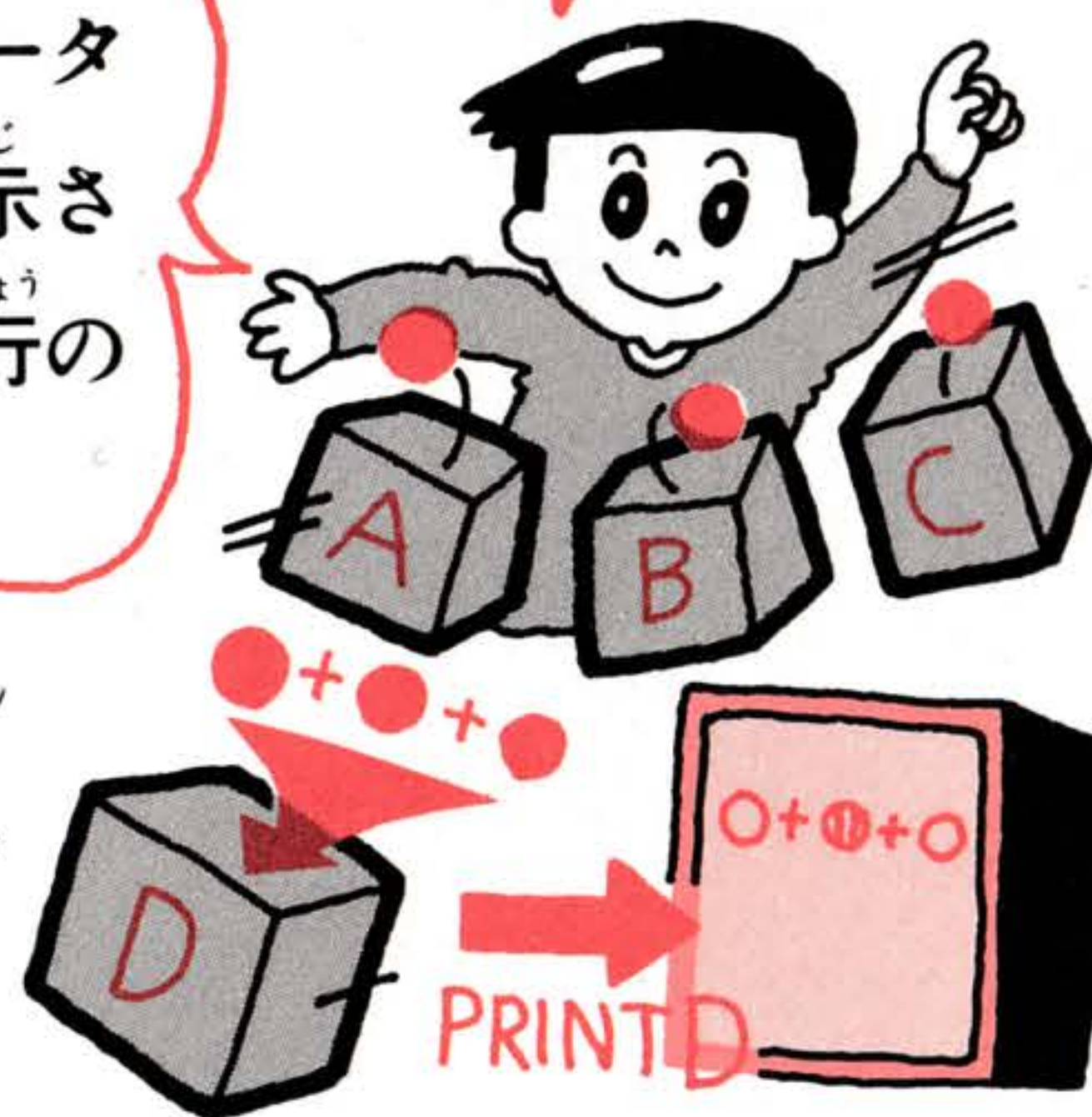
算数の計算では、= (イコール記号) は「等しい」という意味をあらわすけれど、コンピュータでは「代入する」という意味なんだ。

算数のかんたんな計算問題みたいだけど



A、B、Cの3つの箱に入っているデータをたして、Dの箱に入れるんだね

Dの箱のデータを画面に表示させるのが40行のPRINT文だ





おな はこ はい で 同じ箱にデータが入ったり出たり

プログラム中の $=$ が代入する意味だということ
がわかるプログラムをもうひとつあげておこう。

```
10 CLS
20 A=5:B=8:C=10
30 D=A*B:A=C/2
40 PRINT D,A
```

30行では、2つの計
算式を：（コロン）
でつなげているよ



まず自分で変数A、B、C、Dそれぞれの値を
考えてから、プログラムをRUNさせよう。きみの
考えた数値と、画面に表示された値が一致してい
れば、きみは変数がわかったということだよ。

40行の、（コンマ）
は、Dの値とAの値
をはなして表示させ
るための、だ

変数名と変数値

AやBが変数名で、変数に代入される値が変
数値だ。

変数にも名まえをつけてやらないと、コンピ
ュータが判断できない。変数名をつけるには、
いくつかの約束ごとがあるので注意しよう。

1 アルファベットのA～Z、数字の0～9ま
でが使える。

2 数字だけの変数名はつけられない。

3 最初の1文字はアルファベットを使う。

4 2文字以内でつける

5 大文字でも小文字でもよい。

A 1はOK、
1 Aはダメ

BASIC命令
にあるIFやTO
は使えないよ



●変数は変化する数、Aの値はどんどん変わる

20行まではわかるね。
さあ、問題は30行だ。
2つの計算式がある。

$$D = A * B$$

$$A = C / 2$$

Dの箱には、A(5)
とB(8)をかけた数を入
れる。

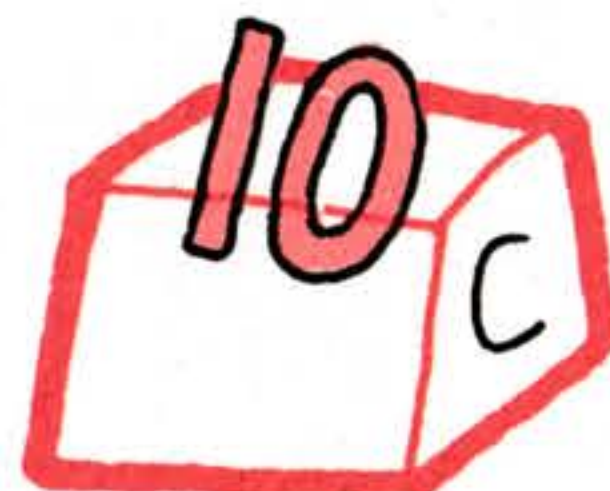
次のAは？ 20行の
5を入れて

$$5 = C / 2$$

なんてやった人はいな
いかな？

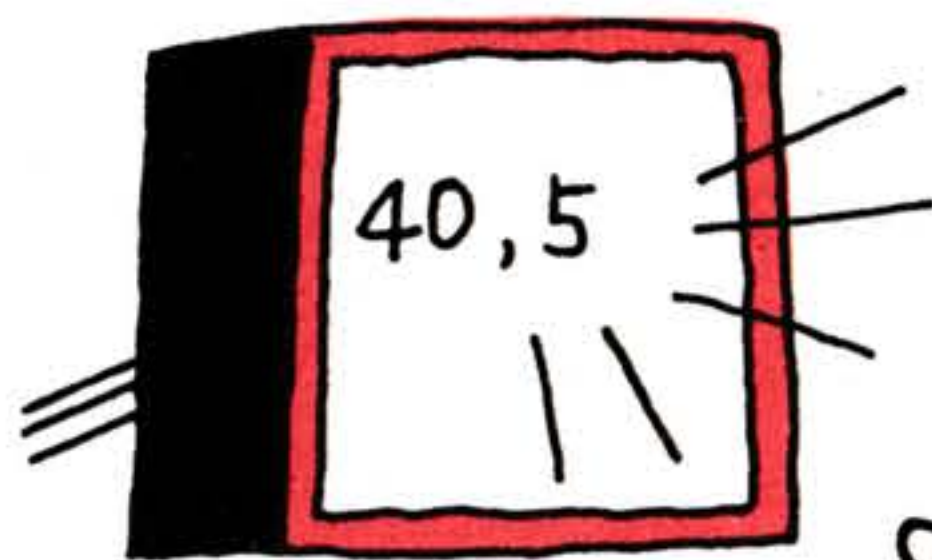
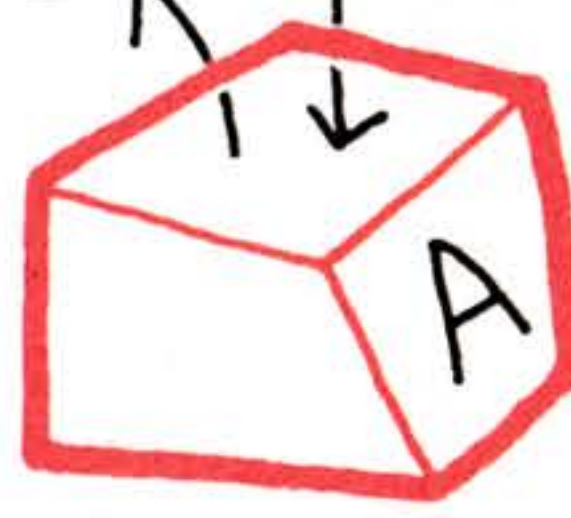
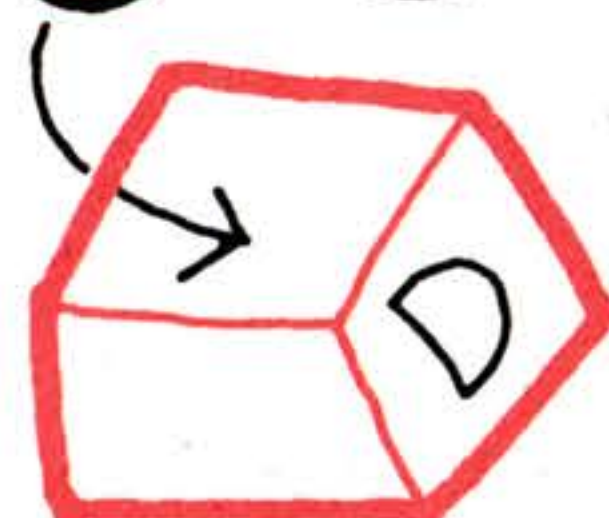
$A = C / 2$ は、Aの
箱にC/2の値を入れ
るという意味だけど、
そうすると、前にAの
箱に入っていた5はど
うなったのだろう。

同じ変数に新しいデ
ータを入れると、古い
データは出てしまうん
だ。つまり、5は出て
しまい、C/2が入る。



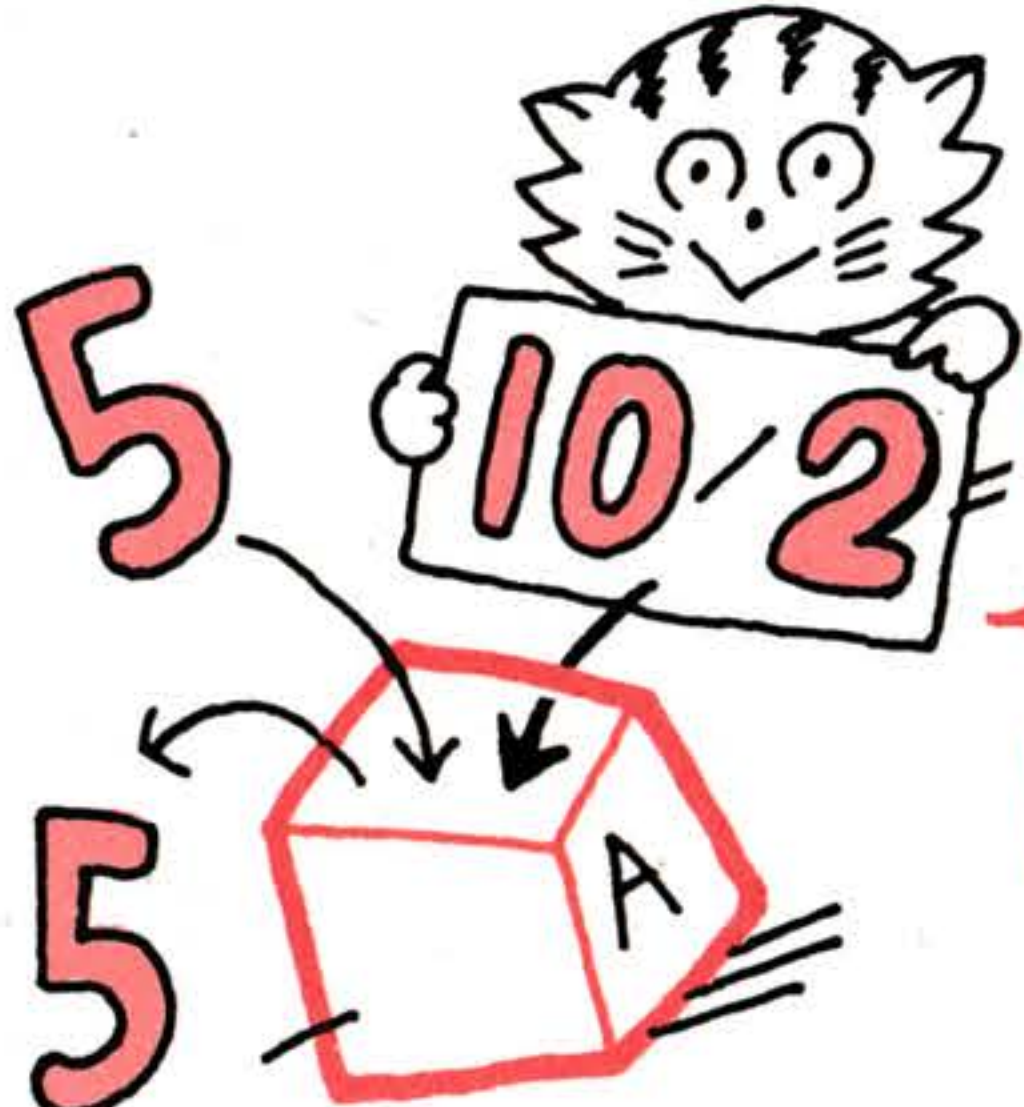
5 * 8

5 10 / 2



このプログラムでは、
B、C、Dの箱(変数)
は一度しか使わないか
ら変数値は変わらない

別の値が入れば
変わるヨ



Aの箱には、データが
出たり入ったりして、
最後に入っているデー
タが変数値になる



文字のデータを入れるときは

文字変数^{も じ へんすう}は変数名^{へんすうめい}のうしろに\$^{ドルマーク}をつける

データには大きくわけて2つの種類^{しゆるい}がある。ひとつは数値データ^{すうち}、もうひとつはことばなどの文字データ^{も じ}だ。

数値データ^{すうち}を入れる箱^{はこ}を、「数値変数^{すうち へんすう}」といい、いま計算問題^{けいさん もんだい}で使った変数^{へんすう}だ。

文字データ^{も じ}を入れる箱^{はこ}は、「文字変数^{も じ へんすう}」または「ストリング変数^{へんすう}」という。名まえはおぼえなくてもいいけれど、次の2つのことを守らなければいけない。

- 1 文字変数^{も じ へんすう}には\$^{ドルマーク}をつけて、数値変数^{すうち へんすう}と区別^{く べつ}する。
- 2 文字変数^{も じ へんすう}に入れるデータはかならず“(ダブルクォーテーション)”で囲^{かこ}む。

たとえば、

10 CLS

20 A\$ = “アタリ!”

30 PRINT A\$

文字変数^{も じ へんすう}に数^{かず}を入れると、コンピュータは数値^{すうち}ではなく、文字^{も じ}だと判断^{はんだん}されることに注意^{ちゆうい}しよう。

5は数値^{すうち}だけど、“5”はただの文字^{も じ}の5だ。

文字変数^{も じ へんすう}もたし算^{さん}ができるけど、計算^{けいさん}するのではなく、文字列^{も じ れつ}と文字列^{も じ れつ}をつなぐんだ。

数値変数^{すうち へんすう}と文字変数^{も じ へんすう}は別のものだから、2つの組み合わせ^{くみあわせ}はできないよ

中は数^{すう}字^じだよ

中は文^も字^じだよ

文字^{も じ}と数字^{すうじ}はたせないよ。ダメな例^{れい}だ

A=A+B\$



●文字変数に代入された数字は数ではなく文字列だ

```
10 CLS
20 A=5:B=7
30 A=A+B
40 PRINT A
50 END
```

```
10 CLS
20 A$="5":B$="7"
30 A$=A$+B$
40 PRINT A$
50 END
```

数値変数と文
字変数のちがい
をみましょう。

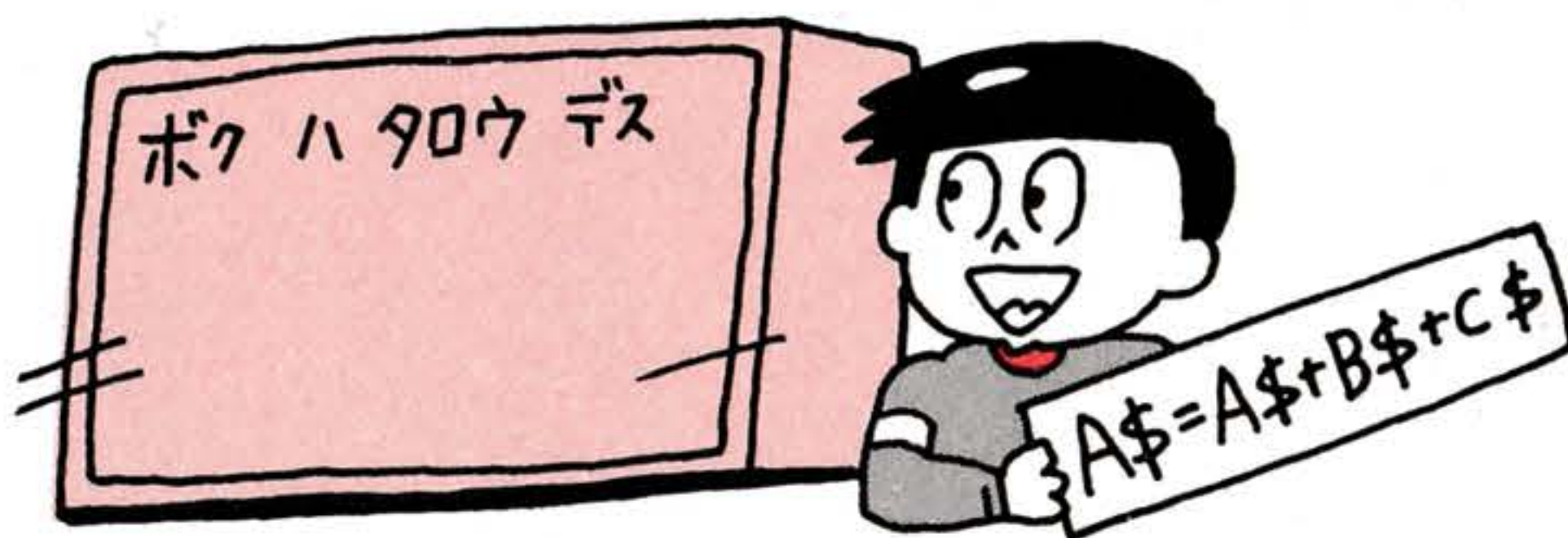
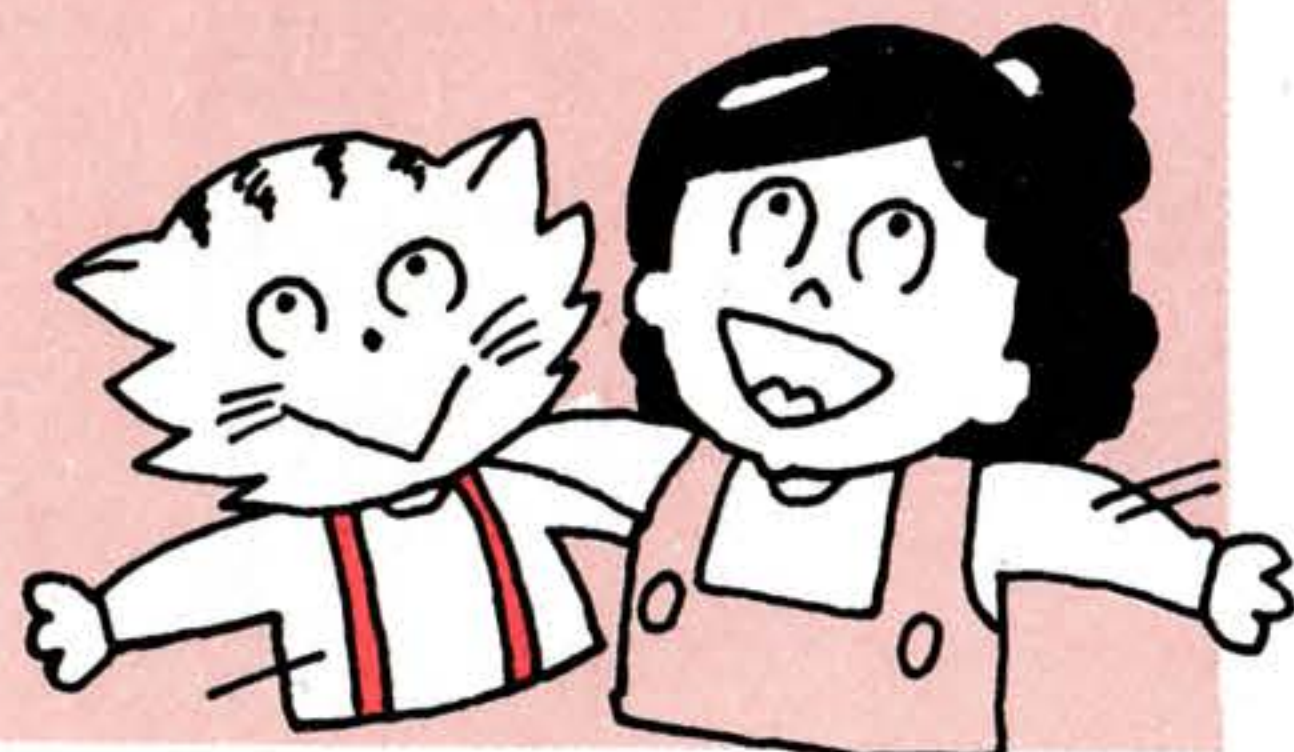
Aの箱に最初に入っ
ていた5とBの箱の7を
たして、もう一度Aの
箱に入れるんだね

えっ、文字をたし算す
るの？

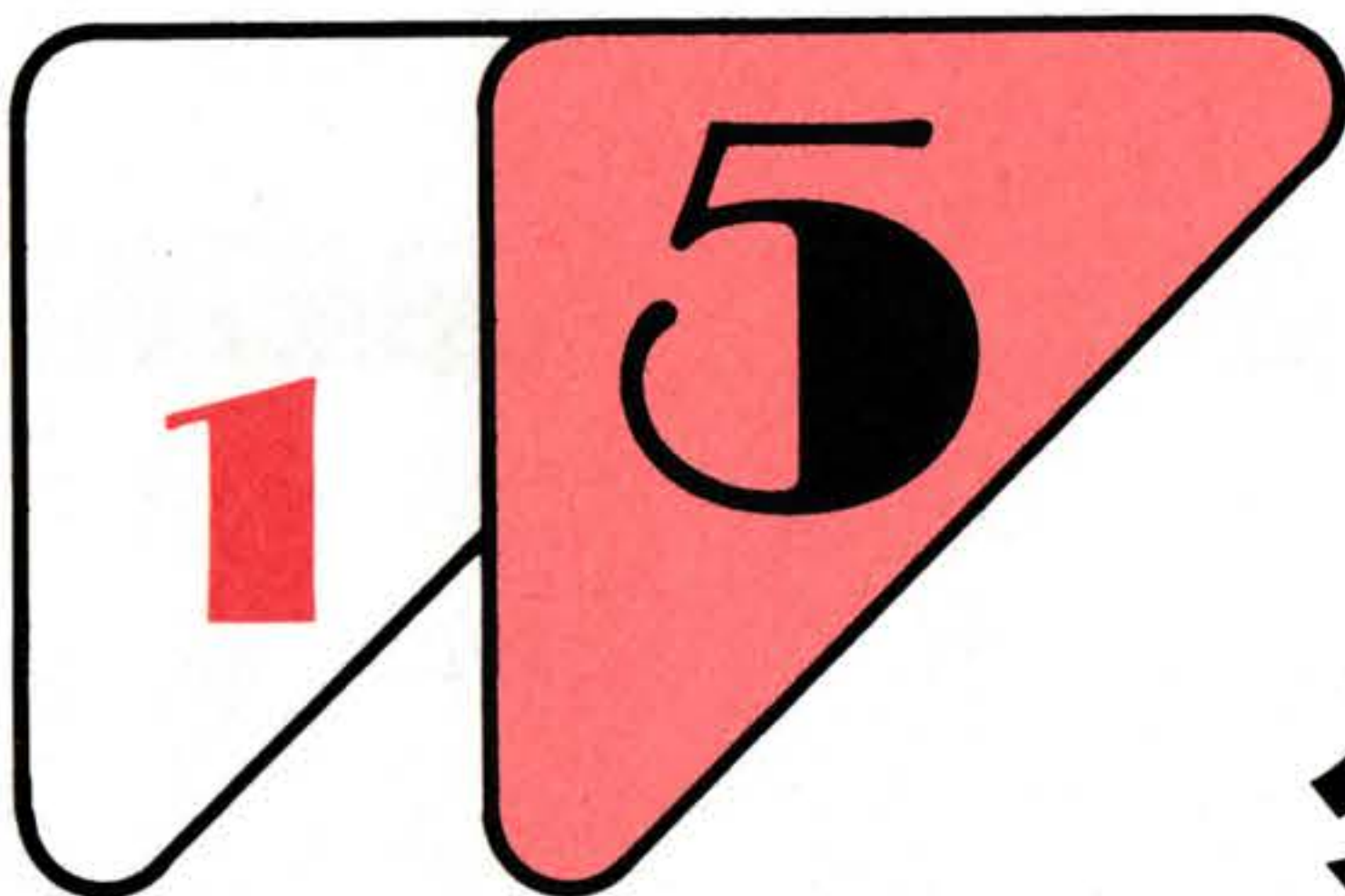
あっ、そうか。文
字と文字をつなぐ
という意味だな



```
10 CLS
20 A$="ホ"
30 B$="タ"
40 C$="テ"
50 A$=A$+B$+C$
60 PRINT A$
70 END
```



文字変数のたし算は、数値変数のよ
うに計算するのではなく、変数に代入
した文字と文字をつなぐことだ。もち
ろん、文字だから、ひいたり、かけた
り、わったりはできない。



クルマを走らせる



表示の位置をどんどん変える

●LOCATE、PRINT、変数のまとめプログラム

```
10 CLS
20 SCREEN 0,0
30 WIDTH 40
40 FOR N=1 TO 35
50 LOCATE N,10:PRINT"  "
60 LOCATE N,11:PRINT"  "
70 LOCATE N,12:PRINT"  "
80 LOCATE N-1,11:PRINT"  "
90 LOCATE N-1,12:PRINT"  "
100 NEXT
```

今まで勉強した命令を使って、画面にクルマを走らせてみよう。

クルマは、LOCATE命令で表示する位置を決め、PRINT文で一と●を組み合わせさせて表示させる。これが50行から70行までのプログラムだ。

80行の " " の中にはなにも書かれていない。これは、車が移動したあとに残る影を消しているんだ。90行の*で、けむりのかんじを出してみた。

横に走らせるということは、画面の横の座標、つまりX座標の上を移動させるということだね。

上下に動かすならY座標の上ということだ。

20行のSCREEN命令は、文字や記号を表示するためのテキストモードを指定しているんだ

ふつうテキストモードのX座標は29文字表示できるようになっているけど、WIDTH 40を指定すると40文字に広がるんだ。それが30行だ

0 → 28

29文字

0 → 39

40文字

WIDTH

このLOCATE命令にあるX座標のNが、クルマを走らせるポイントなんだ。

Nは変数だから、そこにX座標の値を入れてやらなければいけない。40行を見てみよう。

FOR N=1 TO 35

これは、「Nに1から35までの値を入れよ」という意味なんだ。つまり、いちばんはじめのクルマは、50行から90行にあるNに1の値を入れたところに表示される。次のクルマは、Nに2を入れたところに表示される。次は、3、4……と最後にNに35を入れたところに表示される。

これを続けて表示させれば、画面の左から右へクルマを走らせることができるわけだ。

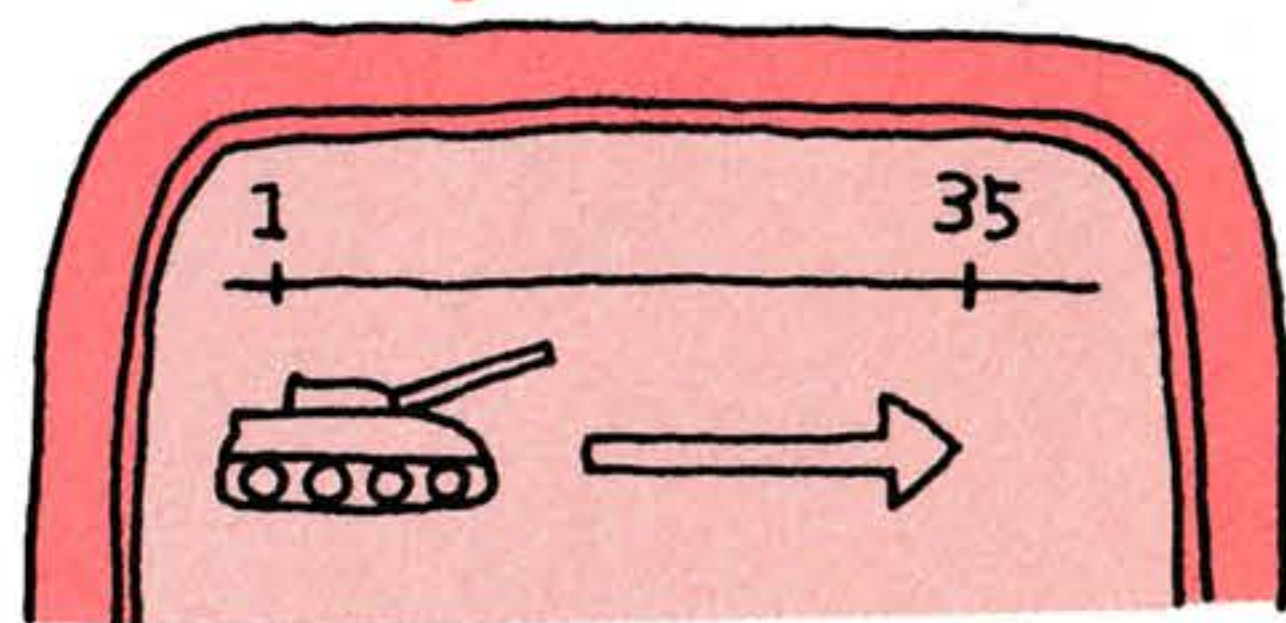
●FORとNEXTのあいだを35になるまでくり返す

X座標の上を1から35まで移動するのだから、Xの値を1から35まで変えて、35個のクルマを書くプログラムを作ってRUNさせれば、走らせることができる。でもそれでは、とても長いプログラムになってしまうね。

これを解決してくれるのが、FOR～NEXT命令なんだ。NEXTは100行にあるね。FORとNEXTには含まれた命令を、決められた回数だけくり返すんだ。このプログラムでは、Nが35になるまで、「クルマを表示する」という同じ仕事をくり返してくれる。ずいぶん便利な命令だね。

クルマの位置より1つ左にけむりを表示させるのだから、けむりの位置はN-1になるね

前に作った戦車をこのプログラムのよう動かしてみよう

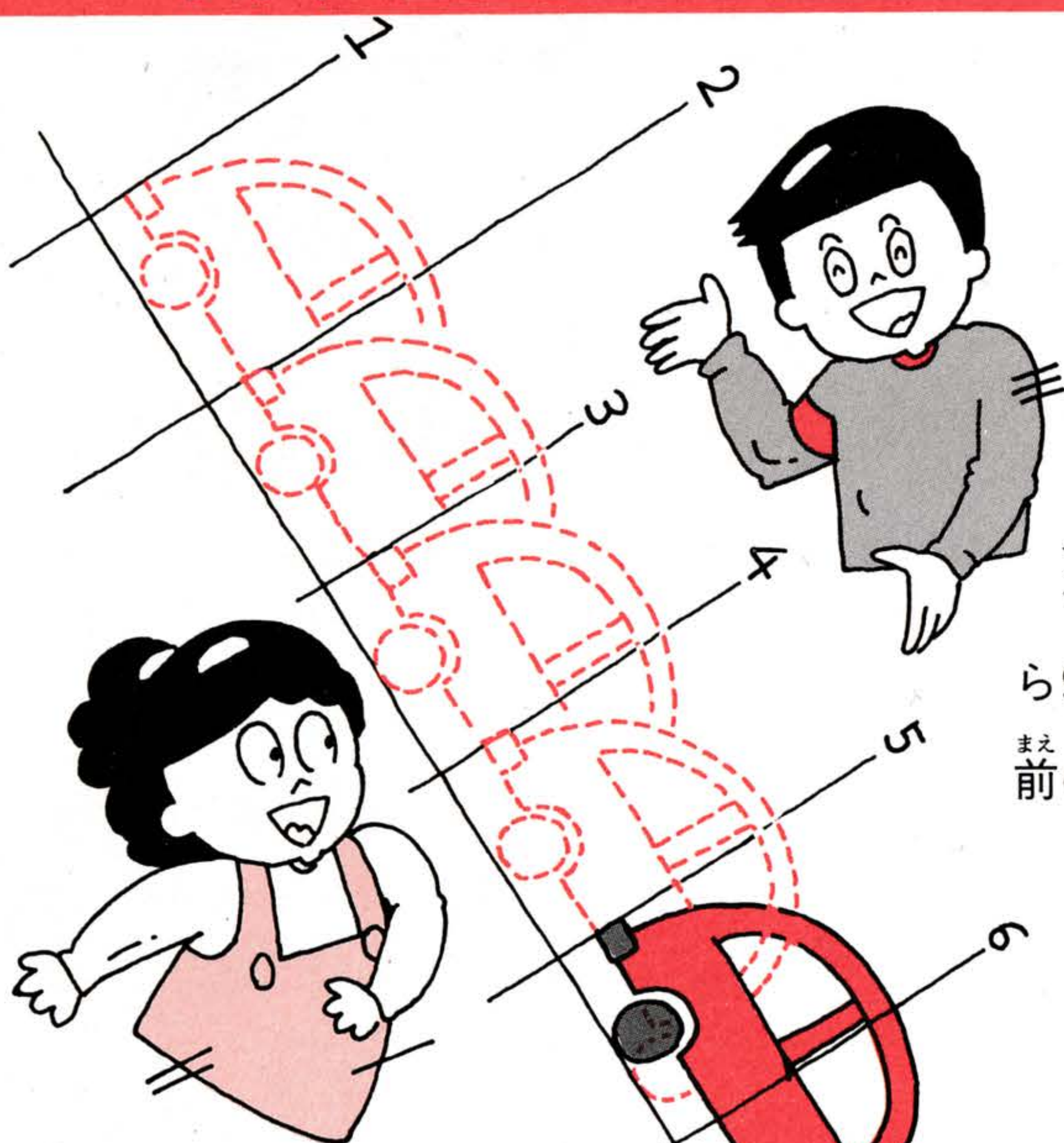


FOR N=1 TO 35

FOR～NEXTのくわしい説明が68ページにあるから、ちょっと読んでみてごらん



●クルマが画面の上を走る、走る、走る



さいしょのクルマの位置は、

Nに1を入れて、

LOCATE 1, 10

LOCATE 1, 11

LOCATE 1, 12

つぎはNに2を入れて、50行から90行のプログラムをくり返す。

前のクルマは消える。

LOCATE 2, 10

LOCATE 2, 11

LOCATE 2, 12

Nに35が入るまで同じ仕事をくり返す。このくり返しの仕事をしてくれるのが、FOR~NEXT命令だ。

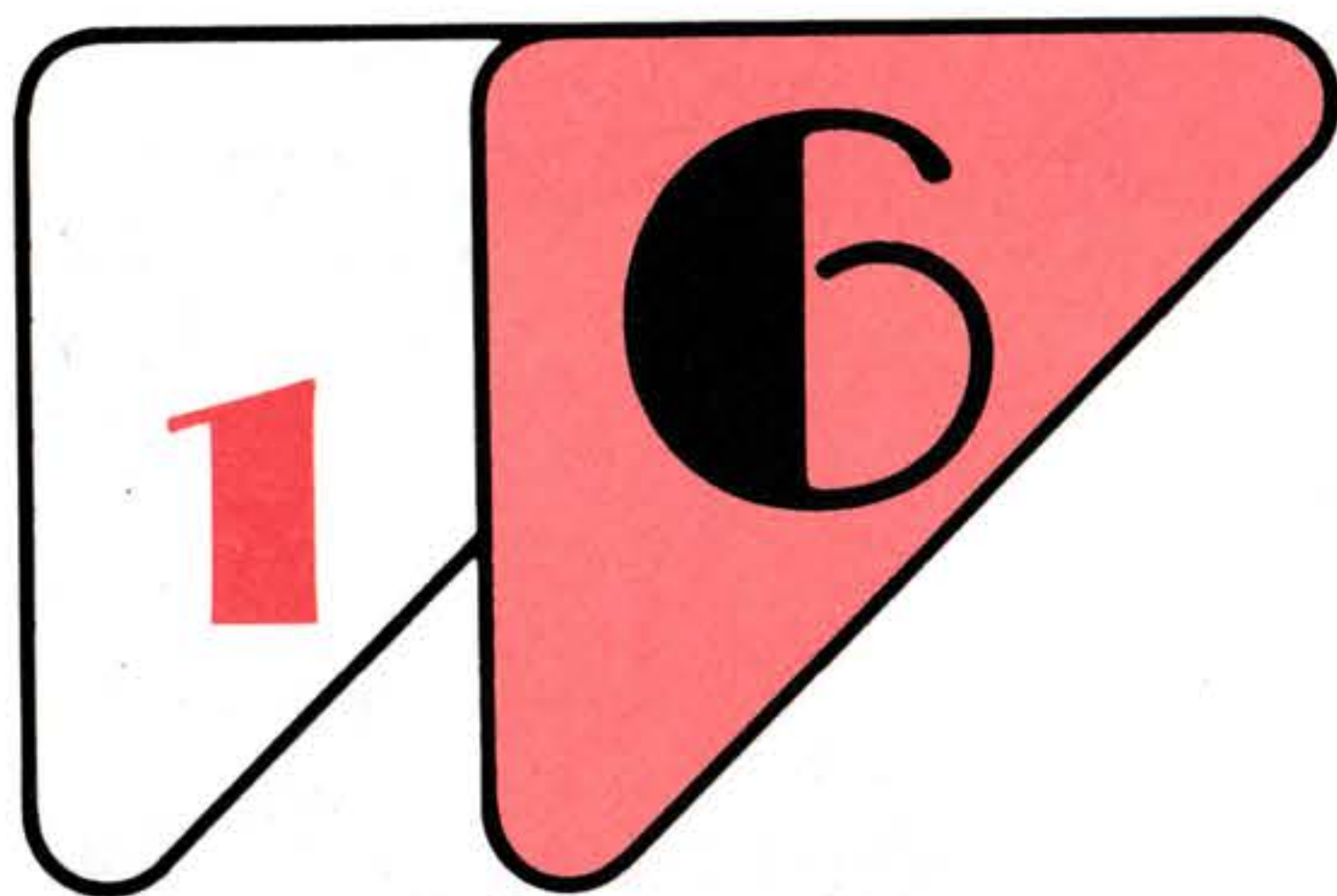
その次は、N

に3を入れる

その次は4

いま表示したクルマを消して、その位置の右どなりに新しいクルマを表示させる。この動作をくり返すと、見た目では、クルマがどんどん右に走っているように見えるんだ





プログラムを 入力する前に



メモリ、行番号、表示文字数

NEW命令 ● メモリの中のプログラムを消してスッキリメモリ

NEW命令は、メモリの中に入っているプログラムを消してやる命令だ。

スイッチを入れたまま、次から次へとプログラムを入力していくと、メモリの中はどんどんいっぱいになって、いろいろなプログラムがまざり合ってしまう。そうすると、新しく入力したプログラムが、正しく実行されないことがあるんだ。

だから、スイッチを入れたままのときは、次のプログラムを入力する前に、**[NEW]****[RETURN]**キーで、記憶されているプログラムを消去したほうがいいよ。

LIST命令を実行して、画面にプログラムがなにも表示されなければ、NEW命令がちゃんと実行されているよ



AUTO命令 ● 行番号を自動的につけてくれる

行番号を自動的につけてくれるのが、AUTO命令だ。

[A]**[U]****[T]****[O]**とキー入力して、**[RETURN]**キーをお押しすると、行番号は自動的に10から始まって、そのあとは、**[RETURN]**キーを押すごとに、10、20、30、40……と10きざみに行番号が付けられる。

プログラミングも
スムーズ



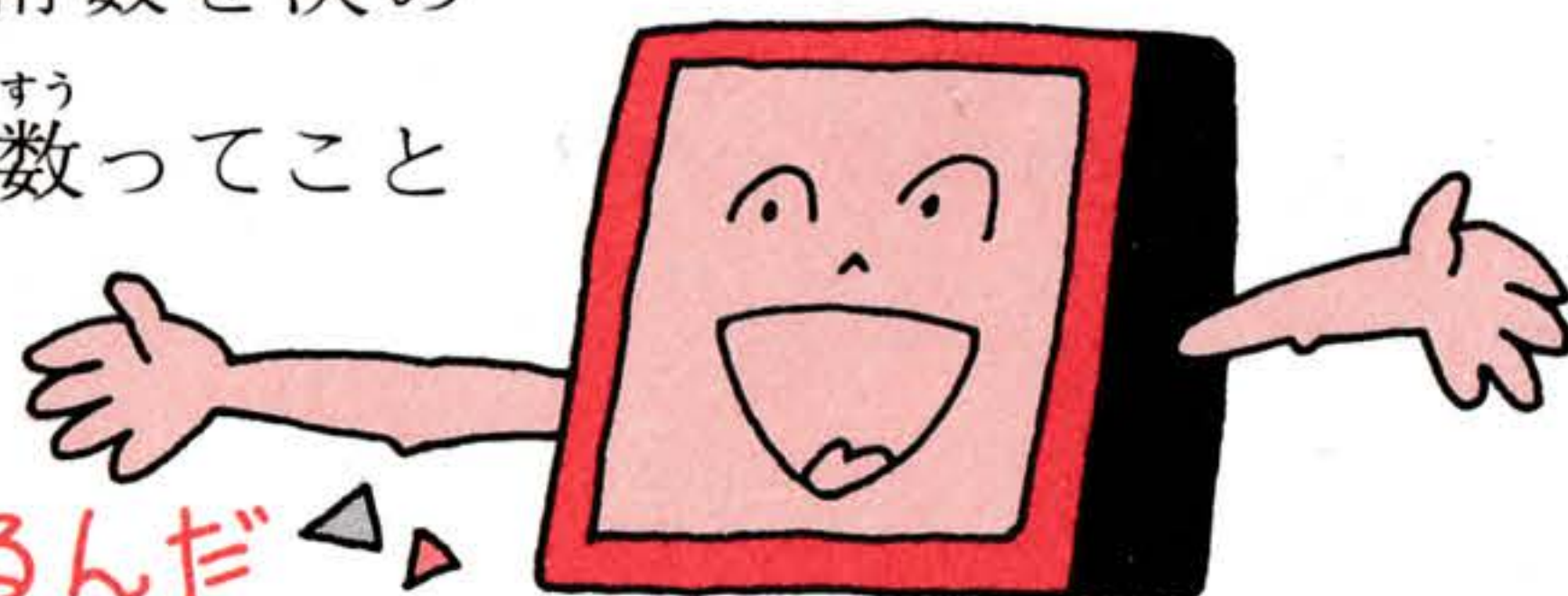
そのほか、出だしの行番号やきざみの数を指定すれば、いろいろな行番号のつけ方もできるよ。

AUTO命令を取り消したいときは、CTRLキーとSTOPキーを同時に押すんだよ。

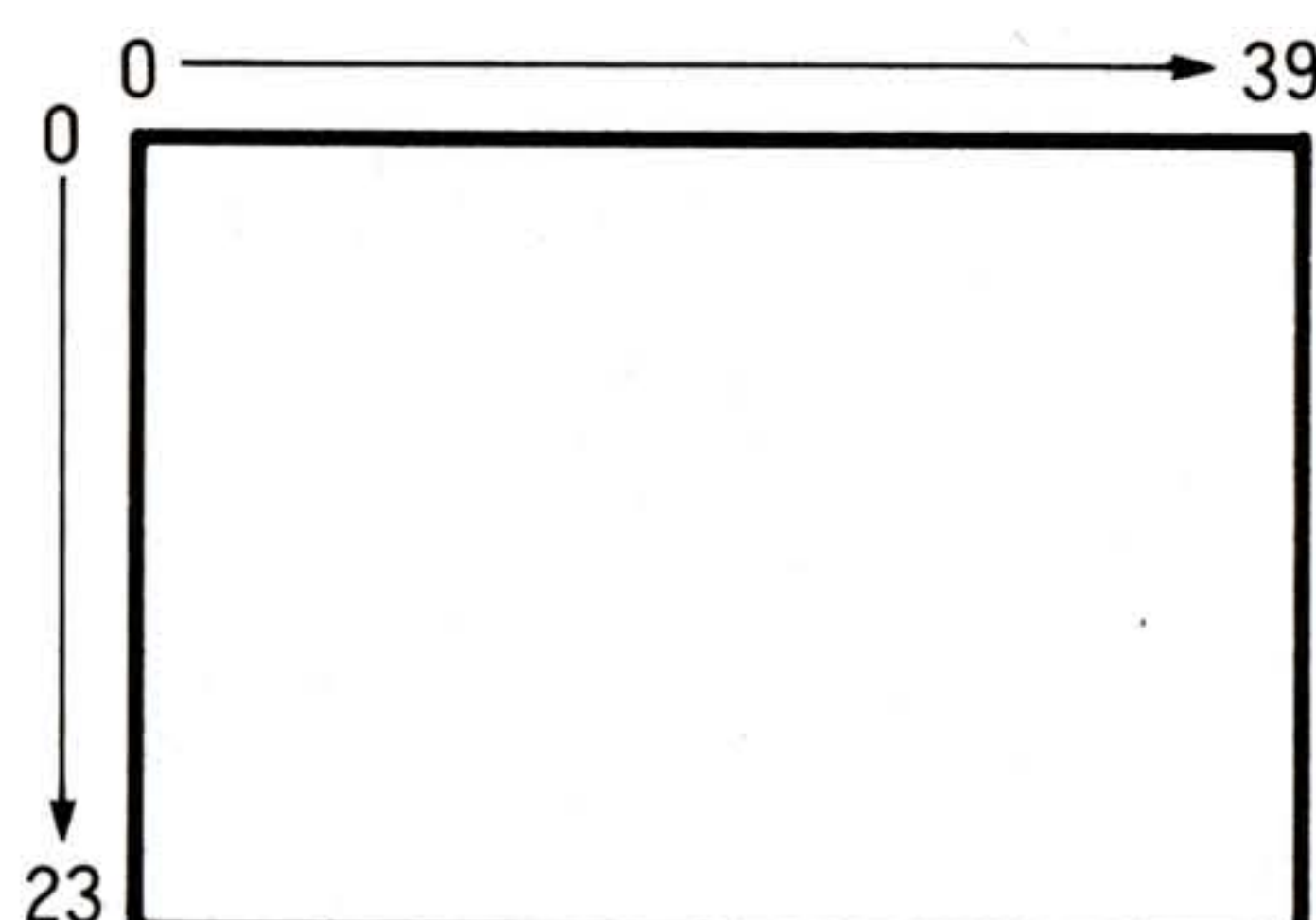
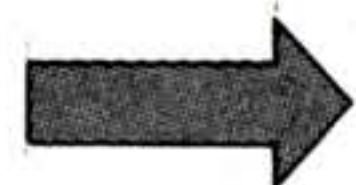
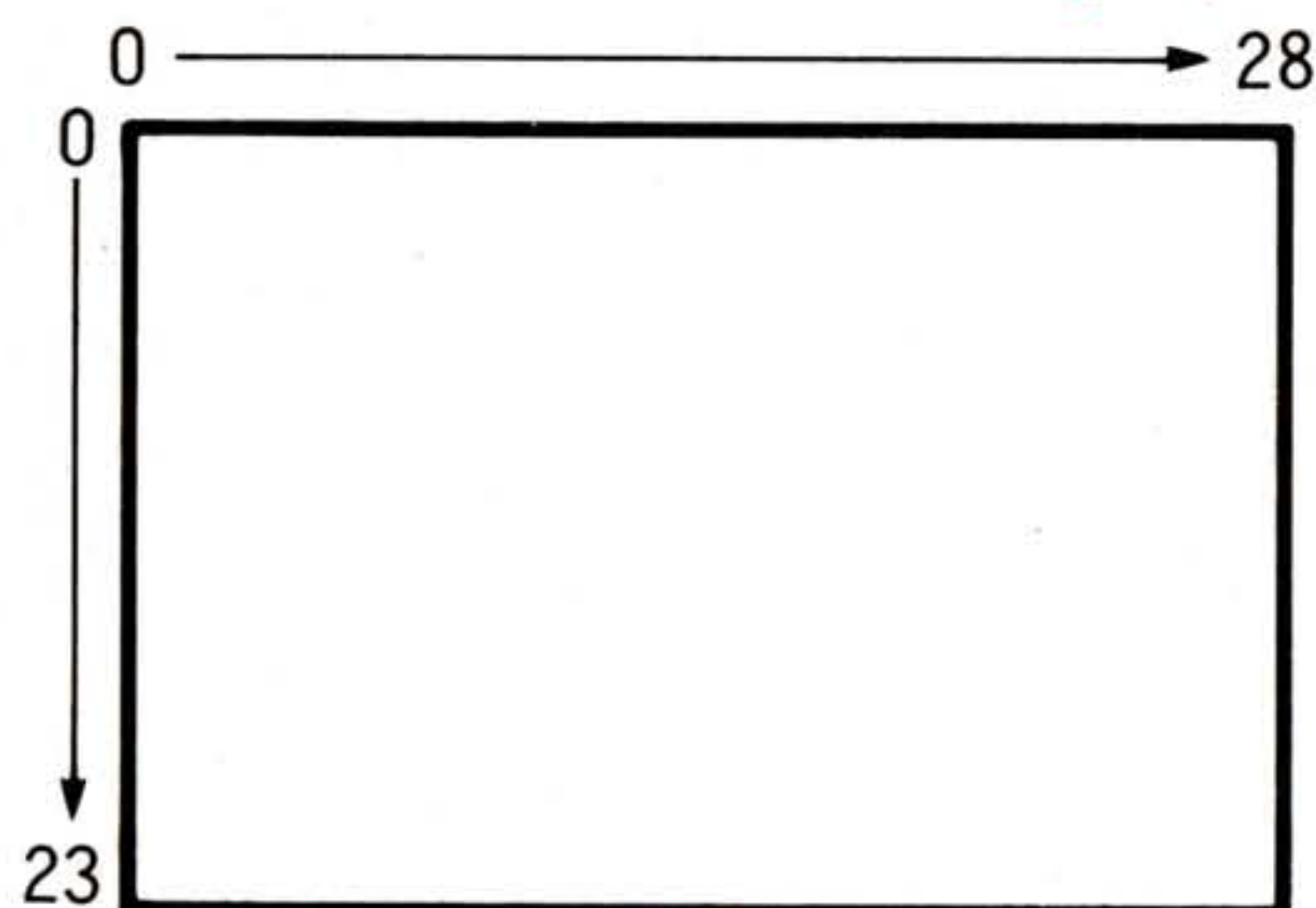


WIDTH命令 ● 画面に表示させる桁数を決める

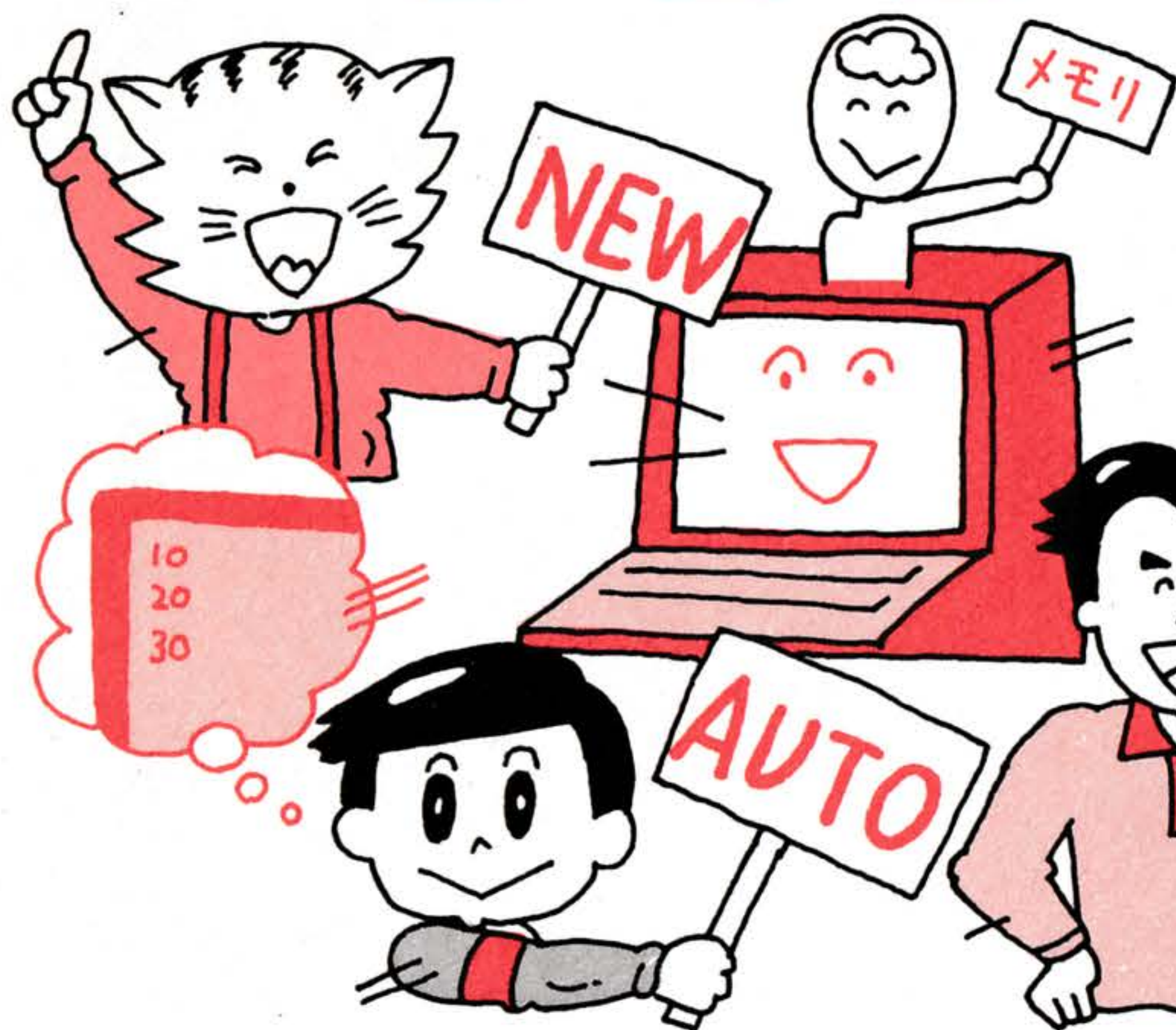
WIDTH命令は、画面に表示させる桁数を決めるんだ。つまり、横に表示できる文字数ってことだね。



画面だってワイドになるんだ



WIDTH 40 を実行すると、横に40文字分、入力できるようになるよ

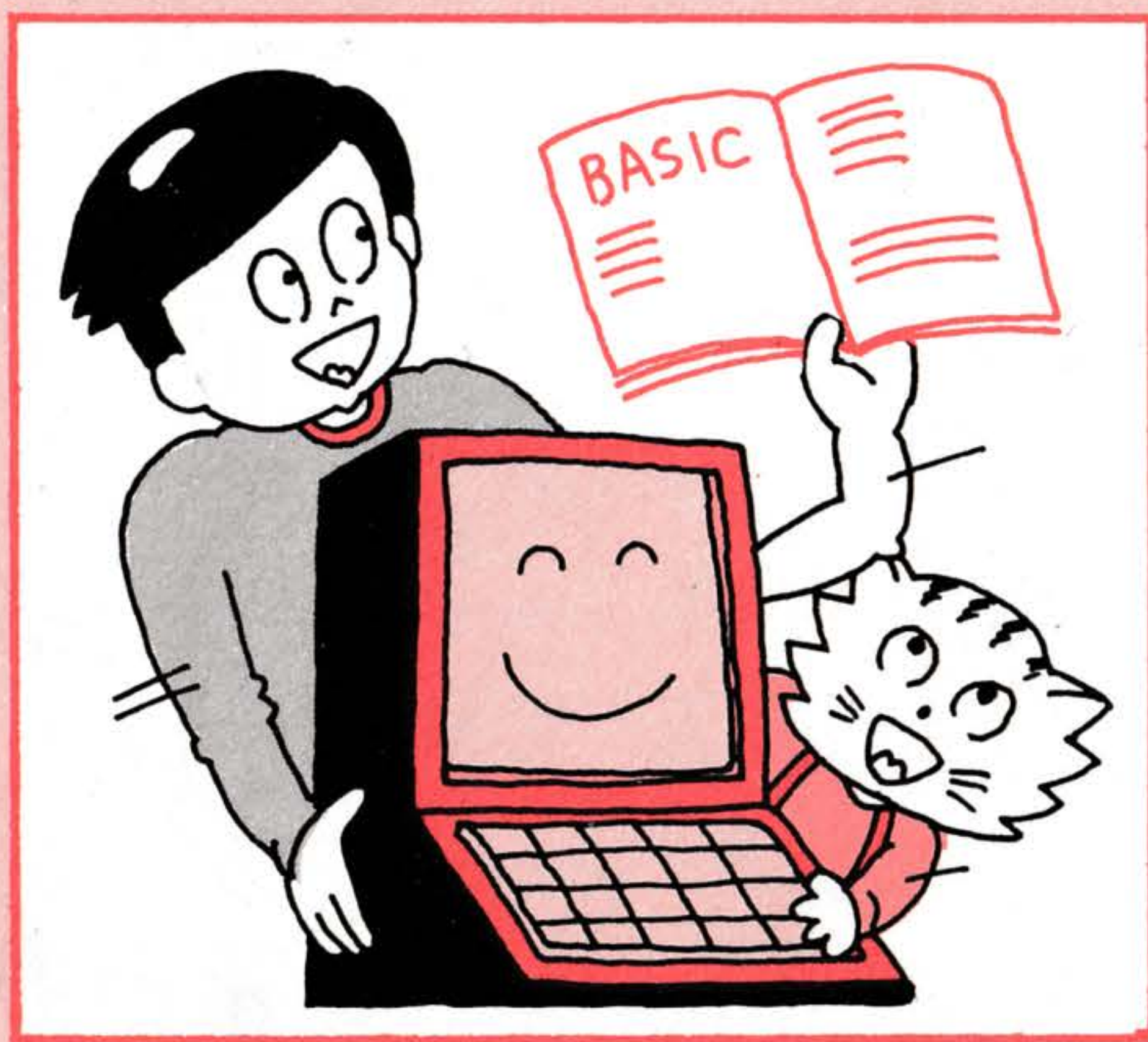


メモリをからにして、行番号がきれいにつくようにAUTO命令を入れて、必要だったらWIDTH命令と。よし、これで入力準備ができたぞ



パート2

BASICを マスターしよう



21

とびとびに プログラムを実行



ゴーツー GOTO 50 なら50行に行く

ゴーツーめいれい GOTO命令 ● 指定された行にジャンプする

プログラムの行番号は知っているよね。プログラムの各行の行頭にならずついていて、最初は小さな数字で、後ろになるにしたがって、だんだん大きな数字になっていく。

そして、パソコンは、行番号の小さなものから順に実行していく——ということになっている。でも、順序を気にかけずに、好きな行番号のところに行ってしまう場合がある。

たとえば、GOTO命令だ。GOTO命令はジャンプの名選手だ。

10 GOTO 100

とすれば、10行の後ろに20行があっても、30行があっても、そこには見向きもしないで、GOTOの次に書かれた100という行にジャンプして行ってしまう。つまり、プログラムの行番号の順序は考えずに、指定した行に飛んで、その行を実行するというわけだ。英語でGOTOは、～へ行けということだから、同じようなものだね。

プログラムにない
行番号を指定すると
エラーになるよ



あまり使いすぎると
プログラムが
見にくくなるよ
ほどほどに使おう



●GOTO文ショートプログラム

```
10 CLS
20 GOTO 40
30 PRINT "ABC"
40 PRINT "MSX"
50 END
```

左のプログラムは、20行のGOTO文で、30行を飛び越えて、40行にジャンプし、40 PRINT "MSX" を実行する。

MSX

だから、画面には "MSX" とだけ表示するが、30行のPRINT "ABC" は飛び越されたので、実行されない。

ABCは
とばされたから
いつまでたっても
表示されないよ

リナンバー じっこう
RENUMを実行すると
ぎょうとう ぎょうばん ごう
行頭の行番号だけでなく
めいれい ぶんちゅう ぎょうばん ごう
命令文中の行番号も
せいり ぎょうばん ごう
整理された行番号に
か
変わるよ
リナンバー せつめい
RENUMの説明は
111ページにある



ゴーツー めいれい GOTO命令

GOTO文につづいて指定した行番号の行にジャンプして、その行の命令を実行する。

GOTO文は、GOとTOの間を1字だけあけて、GO TO と書いてもいいが、2字以上あけて書くと、GOTO文にはならない。

2

ジャンプして ふたたび戻ってくる



ゴ ー サ ブ リ タ ー ン GOSUB~RETURNとおぼえる

ゴ ー サ ブ めいれい リ タ ー ン もど つぎ めいれい
GOSUB命令●RETURNで戻って次の命令から

どこへでも、指定する行へジャンプするのが、
GOTO命令だったね。ただ、GOTO命令はジャン
プして行ったら、行きっぱなし。もとに戻っては
こない。

GOSUB命令は、GOTO命令と同じで、指定の行
にジャンプするが、その行からの命令を実行した
あと、RETURN文があれば、いちどもとのところ
に帰ってきて、GOSUB命令のあった次の命令か
ら実行していくのだ。だから、GOSUB命令はG
OSUB~RETURNと、ひとまとめにして、おぼえ
ておいたほうがいい。

たとえば、

10 GOSUB 100

として、100行へジャンプして、100行から実行し
たあと、

150 RETURN

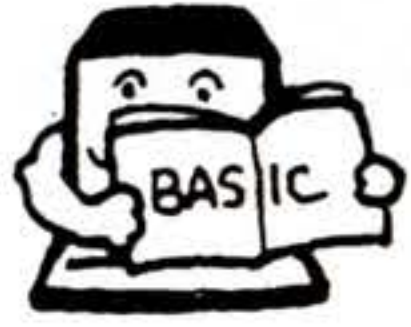
と、RETURN命令があったら、もとに引き返し
て、GOSUB命令の次の20行から実行するわけだ。

ゴ ー サ ブ
GOSUBは
サブルーチンへ行け
リ タ ー ン
RETURNは
もど
戻れということだよ



ゴ ー サ ブ
GOSUB 100は
100行に
ジャンプしろだ





かいちょう いえ かいいん

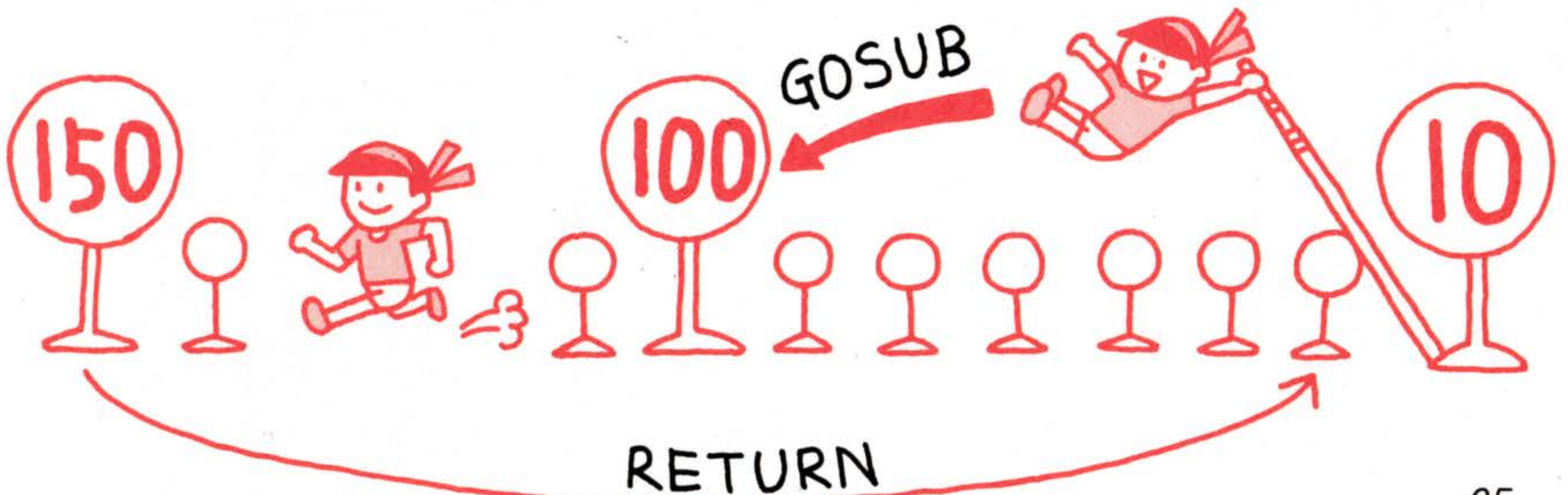
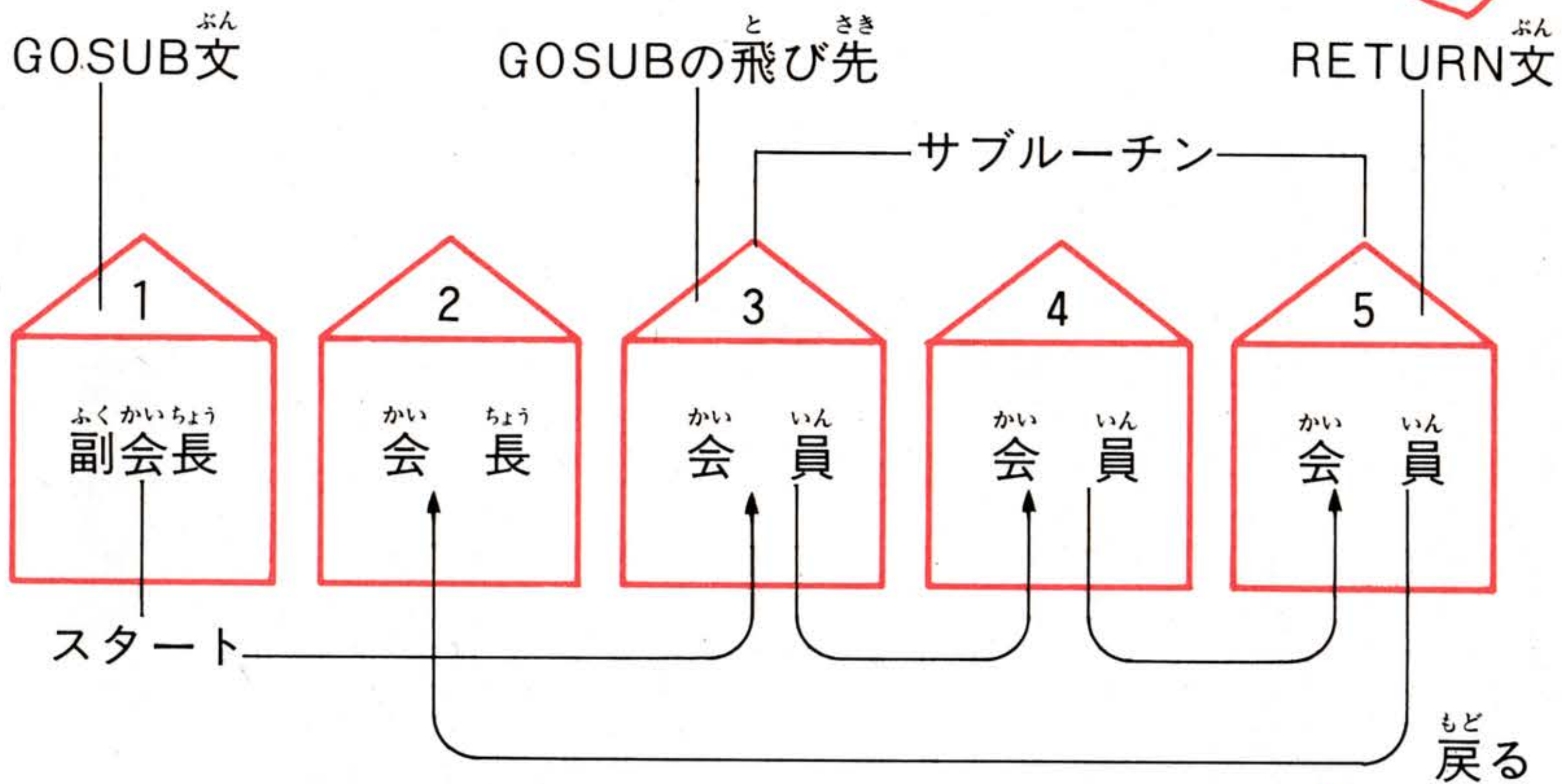
会長の家は会員をまわったあとで

左から1、2、3、4、5と5軒並んだ家があったとしよう。1は町内会の副会長さん、2は会長さん、3、4、5は会員さんだ。

副会長さんが作った文書に全員の名まえを書いてもらい、最後に会長さんの名まえとハンコをもらうとき、どうする？

1から出発して、2は会長さんだからとばして、3、4、5とまわったあと、2の会長さんのところに戻って、名まえとハンコをもらう、という順序だ。

サブルーチンって、
それだけで独立した
プログラムのなかの
小さなプログラム
だよ



●GOSUB文ショートプログラム

した
下のプログラムは、
*ジルシで四角の枠を
か描き、そのなか
に“ワタ
シハ” “MSX” “ヨ
ロシク” と、順番に書
いたり、消したりして
いる。

```
*****
*      ワタシハ      *
*                      *
*                      *
*                      *
*                      *
*****
```



この次は
MSX
その次は
ヨロシク
と書く
はずだ

```
10 CLS
20 GOSUB 110
30 LOCATE 4,3:PRINT"ワタシ ハ"
40 GOSUB 190:CLS
50 GOSUB 110
60 LOCATE 4,3:PRINT"MSX"
70 GOSUB 190:CLS
80 GOSUB 110
90 LOCATE 5,3:PRINT"ヨロシク"
100 GOSUB 190:END
110 PRINT"*****"
120 PRINT"*"
130 PRINT"*"
140 PRINT"*"
150 PRINT"*"
160 PRINT"*"
170 PRINT"*****"
180 RETURN
190 FOR N=1 TO 1000:NEXT N
200 RETURN
```

②カウン
ターのサブ
ルーチン

①四角の
枠を表示す
るサブルー
チン

①のサブルーチンと②のサブルーチンが何度も呼び出される

10→20→110~180→30→40 GOSUB 190→190~200→40 CLS
→50→110~180→60→70 GOSUB 190→190~200→70 CLS……

マルチステートメントに注意。GOSUBの次の命令に戻るんだ。



サブルーチンの使い方

サブルーチンと呼び出すのがGOSUB文だとわりきったほうが、わかりやすい。GO（行け）SUB（サブルーチンへ）だから当然そうなのだ。

サブルーチンは、1個の独立したプログラムだとは、まえにも書いたね。だから、プログラムのなかで、何度も使うような内容があったら、サブルーチンにしておけば、使いたいとき、いつでも、GOSUB文で呼び出すことができ、たいへん便利だね。

それから、1つのサブルーチンの中に、もうひとつサブルーチンを作って、これと呼び出すこともできる。これを、サブルーチンの多重化といって、ちょっと高度なテクニックだよ。

サブルーチンの中に
クリアスクリーンぶん
CLS文を
入れたらいけないよ

サブルーチンを
ゴーツーラン
GOTOやRUNで
実行すれば
エラーだ



ゴースブめいれい GOSUB命令

GOSUBのあとに飛び先の行番号を指定する。

G O S U B 50

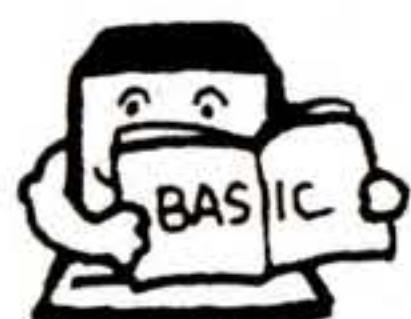
50行に飛んで、50行以降のサブルーチンを実行したあと、RETURN文があったら、GOSUB文の次の命令に戻る。

1つのサブルーチンの中には、複数のRETURN文があってもいい。

RETURN文では、行番号を指定して、特定の行番号にジャンプさせることもできる。



あきずに 同じ動作をくり返せ



たとえば戦車を右に動かしたり

FOR~NEXT ● くり返す動作の回数を指定して

FOR~NEXTは、FORとNEXTには含まれた命令を、指定した回数だけくり返して実行させる命令だ。

FOR N=1 TO 100

として、次にくり返して実行させる命令を書き、

NEXT N

とすれば、FORとNEXTの間の命令を、N回だけくり返す。そして、Nは1から100までだから、Nが1のときに実行、Nが2のとき、3のとき………Nが100のときと、100回実行するということになる。

たとえば、

$$\begin{bmatrix} \# \\ \# \\ \# \end{bmatrix} \text{---}$$

$$\langle \text{***} \rangle$$

と、こんな戦車があったとして、これを画面に表示して動かすなどというときに使える命令だ。

動かすというのは、表示を消して、その隣に同じ表示を描いて、また消して、隣に描いて、消して……という動作をくり返せばいいのだから――。

Nは変数だよ
変数だから
状況しだい
で変化する数だ



FOR~NEXTの

あいだに
戦車を描いて消す
動作を入れればいい



●FOR~NEXT文ショートプログラム

```

10 CLS
20 FOR N=1 TO 20
30 LOCATE N,10
40 PRINT"  [#]--"
50 LOCATE N,11
60 PRINT"  [#]"
70 LOCATE N,12
80 PRINT"<*****>"
90 FOR I=1 TO 200:NEXT
100 LOCATE N,12
110 PRINT"  "
120 NEXT
130 LOCATE N-1,12
140 PRINT"<"
    
```

FOR~NEXT文の中に、

もう1つのFOR~NEXT

文が90行にある。

90行のFOR~NEXT文は、

この行内を1から200まで行
ったり来たりして、時間か
せぎをしているループカウ
ンターだ。

20行のFOR文と120行の
NEXT文のあいだで、

```

      [#]--
      [#]
<*****>
    
```

```

      [#]--
      [#]
<*****>
    
```

という戦車みたいなキャラ
クタを描いて消し、次に1
だけ右の座標に同じものを
描いて消して、これを20回
くり返している。

戦車みたいなキャラクタが右に動く
描いて消して、右に1ずれた座標に
描いて消すから
左から右に動くように見えるのだ





フォーネクストいこうどう FOR~NEXTの入れ子構造

●FOR~NEXTの中にまたFOR~NEXT

FOR~NEXTの中に、いくつでも、FOR~NEXT
文を入れて使うことだってできる。これを入れ子
構造っていうんだ。

例をいくつか、あげておこう

(1) 10 FOR I=0 TO 100

20 FOR J=1 TO 50

...

60 NEXT J

70 NEXT I

(2) 10 FOR I=0 TO 100

20 FOR J=1 TO 50

...

60 NEXT J, I

(3) 10 FOR I=0 TO 100

20 FOR J=1 TO 50

...

60 NEXT

70 NEXT

中に入れる
FOR~NEXTは
完全に外側の
FOR~NEXTの
中に入って

いなければだめだよ

中と外の
FOR~NEXTが
同じところで
終わるなら
NEXT文を1つに

まとめてもいい
変数の順序は
中の変数、外の変数
の順だ

外と中の変数は
かならずちがった
変数を使うこと

NEXT文の変数は省略できるよ

このときのNEXTは

いちばん近くにあるFORのものだ

FOR~NEXT..



●FOR~NEXTのループ回数に注意する

FOR J=10 TO 1.....初期値が最終値

より大きいときは、1回だけ実行される。

FOR J=-10 TO -5.....小さい値か

ら大きい値までの数、つまり-10から-5までの数5回のループをする。

FOR J=10 TO 10.....初期値と最終値

が同じで増分0のときは、1回だけ実行される。

フォー ネクストぶん
FOR~NEXT文

を使うときは

変数の初期値と最終値に注意しよう



FOR~NEXT命令

FOR~NEXT命令の間の命令を、変数で指定した回数だけくり返す。

FORのあとの変数はIでもNでもよいが、NEXTの変数と同じにする。

FOR 変数=初期値 TO 最終値 STEP 増分
NEXT 変数

初期値の数字から最終値の数字までが、くり返す回数。STEP 増分は、初期値の数から最終値の数までのステップを示す。たとえば、

0 TO 10 STEP 2なら、変数0に2プラスして変数2、2プラスして変数4.....変数10と実行する。STEPは省略できる。

STEPを省略したときは、増分は1とみなされる。

24

天気だったら野球 雨だったら勉強だ



条件しだいで行き先を決める

IF~THEN●IF~THEN~ELSE

もし、キミが100点^{てん}とったら、ボクはアンパンを100個^こたべるとか、あした天気^{てんき}がよかったら、サッカーをしようとか、そのときの状況^{じょうきょう}しだいで、あれをやったり、これをしったりするのが、IF文^{イフぶん}だ。

IF文^{ぶん}には、大きく分けて、

IF~THEN

IF~THEN~ELSE

と2とおりの文型^{ぶんけい}がある。

もし(IF)天気^{てんき}だったら、そのときは(THEN)野球^{やきゅう}をしよう、というのがIF~THEN。

もし、あした天気^{てんき}だったら、野球^{やきゅう}をしよう、でも雨^{あめ}だったら(ELSE)勉強^{べんきょう}しよう、というのが、IF~THEN~ELSEなのだ。

IF A = 5 THEN 100

というふう^{つか}に使う。変数^{へんすう}Aが5のときは、プログラムの100行^{ぎょう}に行け^いという意味^{いみ}だ。条件^{じょうけん}を判断^{はんだん}して、それからどうするかというわけだ。

この=は、代入^{だいにゅう}ではなく等しい^{ひと}の意味^{いみ}だよ。



IF~GOTO

という形^{かたち}もあるよ

「Aが5なら

100行^{ぎょう}に行け^い」は

IF A = 5

GOTO 100だ



●IF～THENの使い方

- 2つの数値や文字列の^{すう ち}大小^{も じ れ つ}を^{だい し ょ う}比較^{ひ か く}する。

```
IF A < 100 THEN A = A + 10
```

(Aが100よりちい小さかったら、Aに10をくわ加える)

IF A\$ = "YES" THEN 100

(変数A \$ の内容が "Y E S" だったら、100行に
ジャンプする)

- ^{すう ち}数値が 0 かどうかを ^{はん だん}判断する

IF A<>0 THEN 150 ELSE 200

(Aが0でなかったら150^{ぎょう}行へ、0だったら200^{ぎょう}行に
ジャンプする)

IF $A+B<>0$ THEN 200

(変数A + Bが0でなかったら 200行にジャンプする)

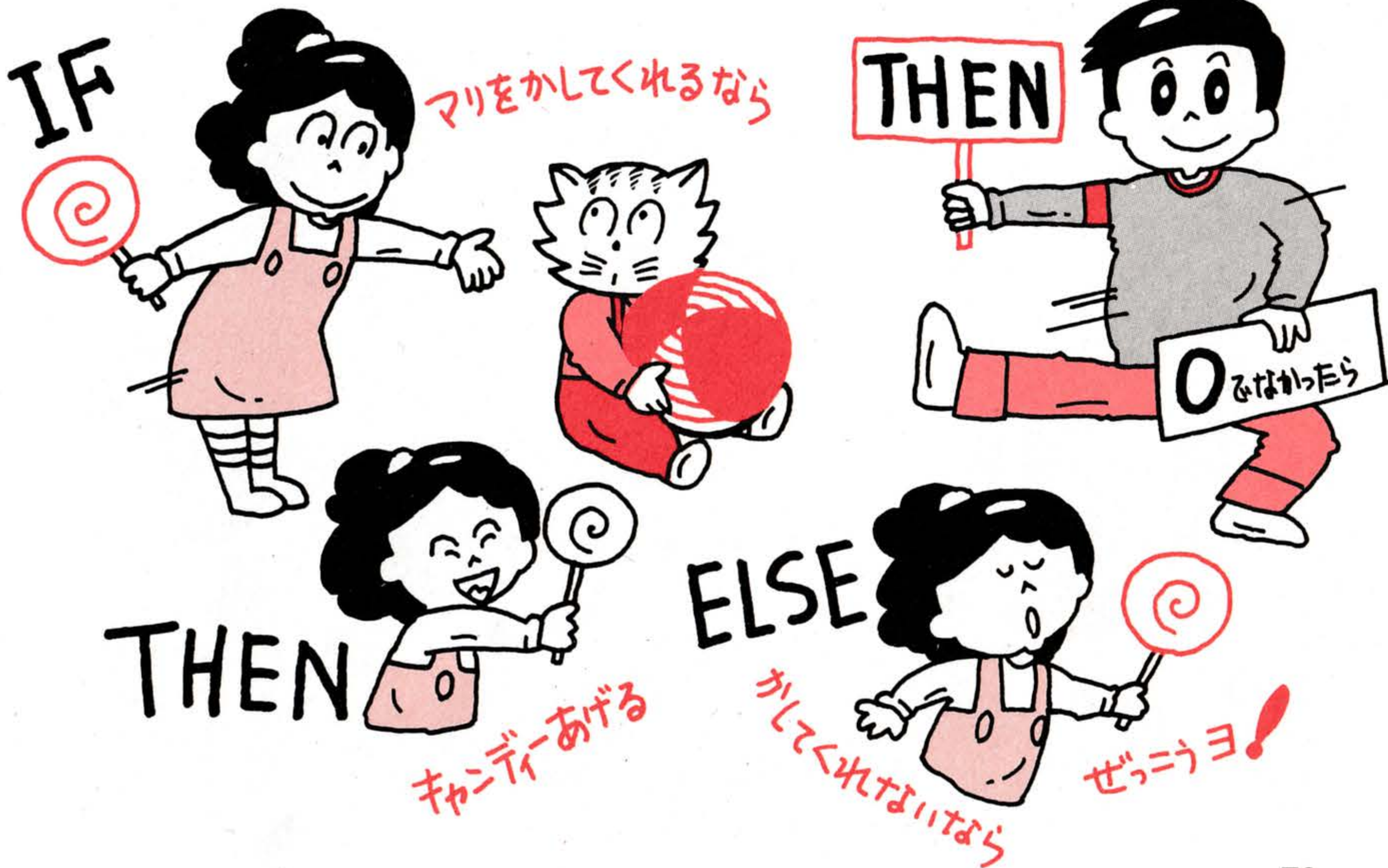
ひ かく えん さん じ 比較演算子の

＝やくや＞をつか
ひかく
比較するのだ



0の判断のとき THENは

0でなかったら
ということなのだね



●IF~THEN文ショートプログラム

```

10 CLS:PRINT"U,H,K,Mノ キーヲ オス"
20 K$=INKEY$:IF K$="" THEN 20
30 IF K$="u" THEN 70
40 IF K$="h" THEN 80
50 IF K$="k" THEN 90
60 IF K$="m" THEN 100 ELSE 20
70 PRINT"ウェ デス":GOTO20
80 PRINT"ヒタリ デス":GOTO20
90 PRINT"ミキ デス":GOTO20
100 PRINT"シタ デス":GOTO20
110 PRINT"シタ デス":GOTO20
    
```

インキーダラー
INKEY\$は

キーボードで
お
押されたキーの
さいしょ
最初の1字を
つかまえるかんすう
関数だ



ABC

まず "U、H、K、Mノキーヲ
オス" と表示されるから、そのと
うりに、まず「U」のキーを押そ
うじゃないか。

U, H, K, Mノ キーヲ オス

ほらUのキーを押したよ
あれ、"ウェ デス" って
いったいこれはなんだ？

ウェ デス



イフ ゼン エルス めいれい
IF~THEN~ELSE 命令

あた じょうけん はんだん つぎ い さき き
与えられた条件がどうなのかを判断して、次の行き先を決める。

IF 条件 THEN 文または行番号 ELSE 文また
は行番号

あた じょうけん あ ぶん ぎょうばんごう じつ
与えられた条件に合っていたら、THENのあとの文または行番号を実
こう 行し、そうでなかったら、ELSEの次を実行する。

「ウエデス」っていったいなんだろう。わかっている人は、スルドイぞ。

U、H、K、Mのキーボードの配置を見てほしい。わかったね。

4文字のうち、**U**は、上にあるではないか。

Hを押してみよう。表示は、“ヒダリデス”だね。**K**を押せば“ミギデス” **M**を押せば“シタデス”だ。

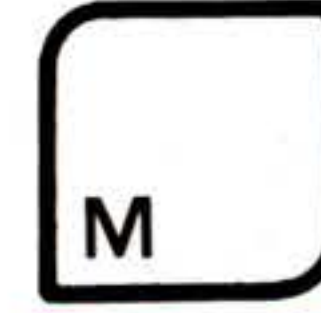
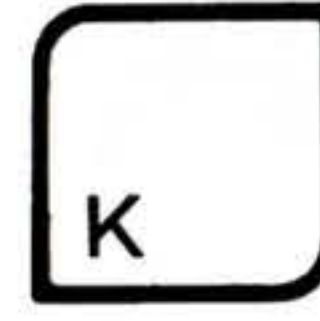
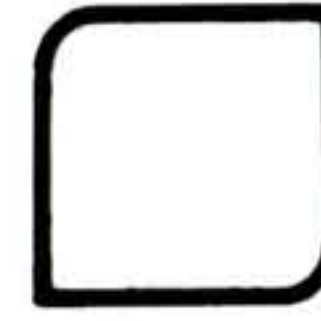
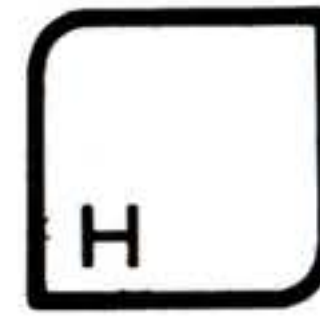
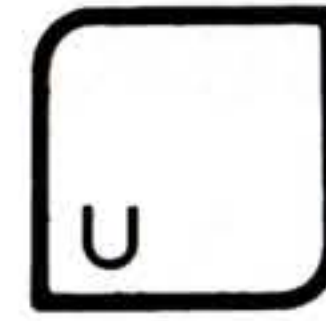
プログラムを見てみよう。

20行 キーが押されなかったら、プログラムがストップして、永遠のキー入力待ちだ。

30行 **U**のキーが押されたら、

(変数K\$が“u”なら)、70行にジャンプ。70行で“ウエデス”と表示して、また20行に戻って、キー入力待ちだ。

40行 **K**が押されたら……、**U**と同じだ。80行で“ヒダリデス”と表示して、20行に戻る。あとは、これと同じくり返した。



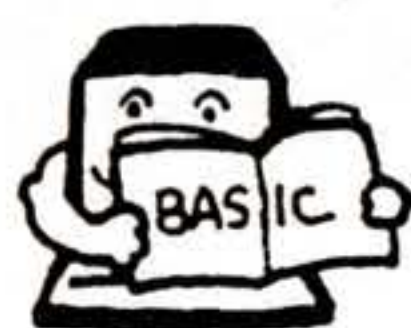
4個のキーのうち、Uのキーは上にある。Hが左で、Kは右、Mは下なのだ

ヒダリ デス

ミギ デス

シタ デス





てん じゆう え か 点でつづる自由なお絵描き

● IF~THEN文のお遊びプログラム

```
10 CLS:KEY OFF
20 SCREEN 0
30 WIDTH 40
40 COLOR 15,5
50 X=10:Y=10
60 LOCATE X,Y
70 PRINT"*";
80 ST=STICK(0)
90 A=0:B=0
100 IF ST=1 THEN A=0:B=-1
110 IF ST=5 THEN A=0:B=1
120 IF ST=7 THEN A=-1:B=0
130 IF ST=3 THEN A=1:B=0
140 LOCATE X,Y
150 PRINT". ";
160 IF X+A<0 OR X+A>39 THEN A=0
170 IF Y+B<0 OR Y+B>22 THEN B=0
180 X=X+A:Y=Y+B
190 LOCATE X,Y
200 PRINT"*";
210 GOTO 80
```

RUNさせると、画面に＊ジ
ルシ（アスタリ
スク）が表示さ
れるから、これ
を画面いっぱい
前後左右にとび
はねさせるのだ。
＊を動かすキー
はカーソルキー。
＊が動きまわ
ったあとには、
「・」が残るから、
どんなおもしろ

い図でも、おもしろくない図でも、きみの腕しだいで描ける。

100行からのIF文にあるST=1、ST=5、ST=7、ST=3は、
＊の進む方向、上下左右を条件にしている。



ひと筆描きのよう
に描けばいいのだな



25

並んだデータを 順番どおりに呼び出す



変数の順とデータの順は一致する

READ~DATA ● 2つそろっていちにんまえ

READ文は、DATA文に書かれたたくさんのデータを呼び出して、READ文にある変数に順序よく代入する命令だ。

READ文があったら、かならずDATAがあるのだから、READ~DATAと、ひとまとめにして、おぼえておこう。

次のプログラムを入力して、RUNさせよう。

```
10 READ A
20 READ B
30 PRINT "A=" ; A
40 PRINT "B=" ; B
50 DATA 8
60 DATA 20
```

A = 8

B = 20

OK



READ文の
最初の変数は
DATA文の

最初のデータを
呼び込むのだよ



データ 8 が
変数 A に代入

データ 20 が
変数 B に代入され
ていくんだって

10行のREAD文にある変数Aに、50行のDATA文のデータ8（定数）が代入され、20行のREAD文にある変数Bに、60行のデータ20が代入されたんだよ、ということを、画面の表示が示しているのさ。





筆が変数なら色がデータだ

●虹の色を塗る順番も決まっている

READ~DATAの使い方を、もう少しくだいて説明してみよう。

絵の具箱がある。いろいろな色の絵の具がある。

虹を描いてみようかな、とおもう。虹の7色は、上から、赤、オレンジ、黄、緑、水色、青、紫の順だ。

色がまじるときたなくなるから、7本の筆を用意して、1本の筆で1色の色を塗ろう。1の筆は赤用だ。2の筆はオレンジ用だ。3の筆は……。

READ文に書く変数が筆だ。

DATA文に書くデータが絵の具の色だ。

書かれた変数の順番に合ったデータが、順番どおりに、それぞれの変数に代入されていく。

READ A, B, C, D...

READ文の変数は
コンマ(,)で区切って
たくさん
書いてもいい



●READ~DATA文ショートプログラム

●棒グラフを描いてみよう

```

100 REM ホウ グラフ
110 CLS
120 SCREEN 0,0:WIDTH 40
130 FOR N=5 TO 17 STEP 2
140 H=H+1
150 READ A
160 LOCATE 3,N:PRINT H;" ":";"
170 FOR I=1 TO A
180 PRINT "*"
190 NEXT I
200 NEXT N
210 DATA 15,8,19,14
220 DATA 9,26,16
    
```

フォー ネクスト
FOR~NEXTは

130~200行

ここで7回まわって

データを7個

ひろってくる

リード ぶん へんすう
READ文の変数Aに

210、220行の

データが7個

順番に入ってくる

1:*****

2:*****

3:*****

4:*****

5:*****

6:*****

7:*****



140 H=H+1

7項目の数が1から1ずつふえて
160行のPRINT文で
書かれていくよ



160 LOCATE 3,N:PRINT H;" ":";"

座標(3, N)の位置から

1行とばしに

「:」ジルシを

表示せよ、だ



●READ~DATAのかたち

☆50 READ A \$, B, C

(代入)

100 DATA タロウ, 175, 68

☆50 READ A \$, B, C

(代入)

100 DATA タロウ, 175, 68

110 DATA ハナコ, 160, 51

120 DATA マリコ, 162, 50

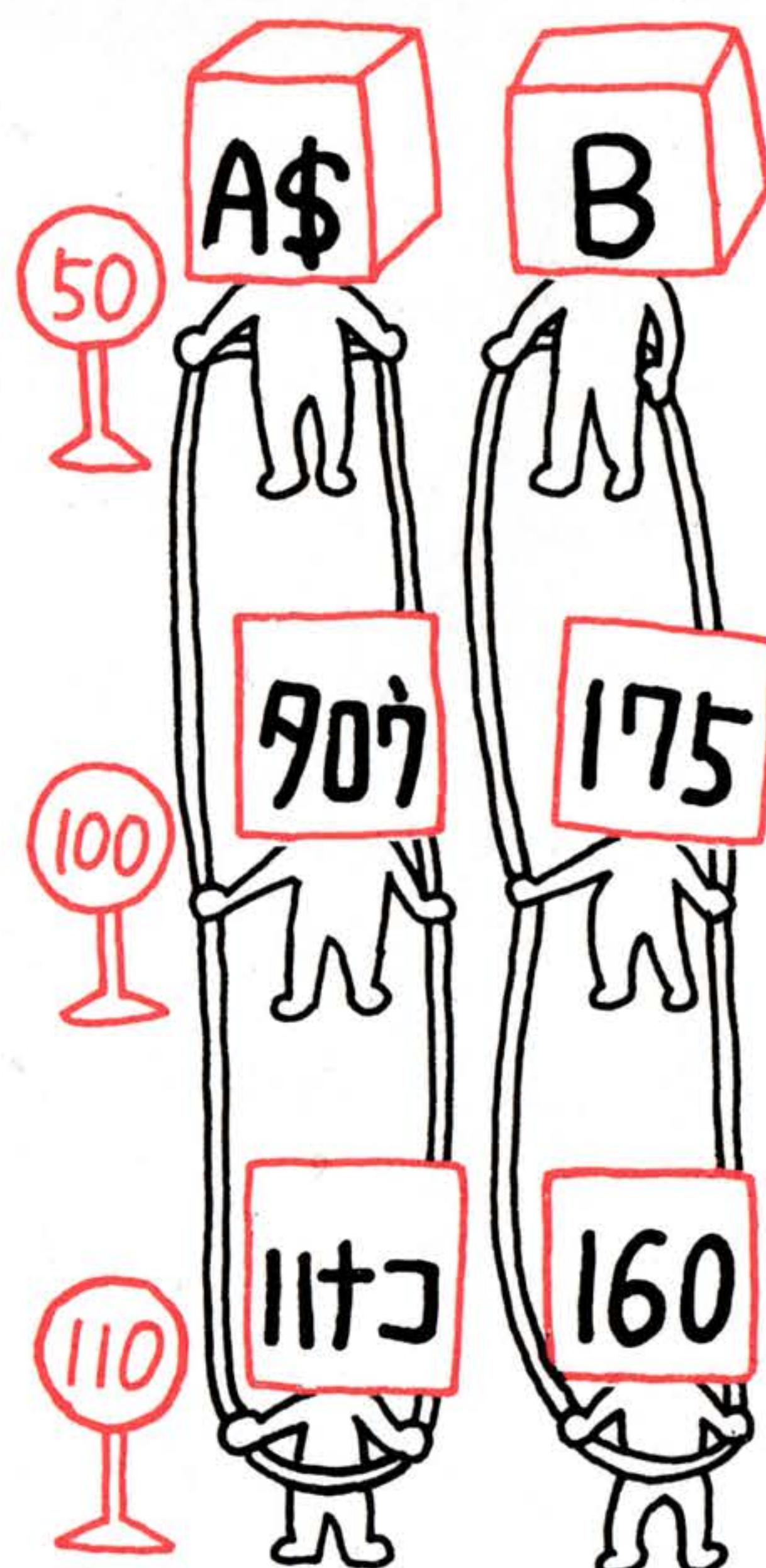
☆50 READ A \$

60 READ B

(代入)

100 DATA タロウ

110 DATA 175



A \$は文字型変数、B、Cは数値変数だ。文字データと呼ぶのは文字変数、数値データと呼ぶのは数値変数なのだ

リード データめいれい READ~DATA命令

READ文は、かならずDATA文がついてくる。

READ 変数, 変数, 変数.....

DATA データ, データ, データ.....

という文型だが、このDATA文のデータが、READ文の変数に、順番どおりに代入される。READ文で使う変数は、数値変数でも文字変数でもいいが、読み込むデータと同じ型でなければ Syntax error となる。



リード データ おうよう READ~DATAを応用しよう

●電話管理プログラム

READ~DATAを使った^{つか}応用^{おうよう}プログラムを組^くんでみた。電話^{でんわ}管理^{かんり}プログラムだ。

プログラムを^{みじか}短くするために、画面^{がめん}の^{おお}大きさを^{ういどす}WIDTH 20 と^{ちい}小さくしているが、^{じつよう}ほぼ、^{ちか}実用に近いプログラムだ。

データをふやして
かくちょう
拡張すれば
でんわちょうが
電話帳代わりの
じつよう
実用ソフトができる



```

10 CLS:SCREEN 0:WIDTH 20
20 DIM A$(5),B$(5):KEY OFF
30 FOR N=1 TO 5
40 READ A$(N),B$(N)
50 NEXT
60 CLS:PRINT" #### TEL カリ ####"
70 LOCATE 0,5
80 PRINT"*****"
90 FOR I=1 TO 5
100 PRINT"*                *"
110 NEXT
120 PRINT"*****"
130 LOCATE 0,15
140 INPUT A$:FOR N=1 TO 5
150 IF A#=A$(N) THEN 180
160 NEXT:LOCATE 0,15
170 PRINT SPC(20):GOTO 130
180 LOCATE 4,7:PRINT A$(N)
190 LOCATE 4,9:PRINT B$(N)
200 LOCATE 2,15:PRINT"キー ラ オシテ クダサイ"
210 S$=INKEY$:IF S$="" THEN 210
220 GOTO 60
230 DATA コクマ,303-8909,ノジマ,876-9500
240 DATA シンセイ,831-0743,セキクチ,800-1234
250 DATA ヤマシタ,42-0820
    
```


●プログラムの使い方

プログラムをキー入力して、RUNさせたら、電話番号を調べたい人の名まえを入力する。その人の番号が、この電話帳に掲載されていれば、名まえと電話番号が表示されるよ。

もちろん、電話番号が登録されていない人の名まえを入力しても、それは表示されない。

電話番号を検索して表示したあと、「キーヲオシテクダサイ」と表示されるから、もう1度してみたかったら、どのキーでもいいから押せば、またスタートする。

●プログラムのかんたんな説明

・データの呼び込み 30～50行のREAD文で、230～250の行のデータを呼んでいる。

・名まえがあるかどうか 入力された名まえのデータがあるかどうかを判断するのは140～160行のFOR～NEXT文だ。名まえがあれば、180行に行つて、名まえと電話番号を表示する。

ヤマシタ 42-0820

・データをふやすなら 新しくデータ文で名まえと電話番号を追加し、20行のA\$(5)、B\$(5)、30行の5、140行の5とある数字の5をデータの数字に合わせてふやせばいい。

このまま
調べられるのは
コグマとノジマ
シンセイにセキグチ
ヤマシタだけだよ

170行は
空白を20個
書いているんだよ
空白を20個書くから
表示された字が
消えてしまう





まちがいの意味と 行番号を表示



よくあるのは^{シンタックス エラー}Syntax error

エラーメッセージ●プログラムを直せばいい^{なお}

^{ひと}人はまちがいをおかしたら、つぐないをしな
ければならない。でも、まちがったかどうかを知ら
せてくれなければわからない。この「オーイ！ま
ちがっているよーん」と^{おし}教えてくれるのが、エラ
ーメッセージだ。

せっかく^{つく}作ったプログラムを^{にゅうりょく}入力して、RUN
させはしたのだが……、画面には^{がめん}無情にも^{むじょう}Synt-
ax errorの^{ひょうじ}表示。このエラーは、^{しょしんしゃ}初心者ほどおか
すエラーだ。

Syntax error in 10

Syntax errorは、^{にゅうりょく}入力したプログラムが、まち
がって^か書かれていたというエラーだ。^{しょしんしゃ}初心者のう
ちは、だれだってやることだから、はずかしがら
なくたっていい。

L I S T RETURN として、プログラムを^{ひょう}表
示させ、まちがいを^{なお}直せばOKだ。

in 10っていうのは
^{ぎょう}10行にまちがいがある
ということだ

^{ぎょう}10行にまちがい
だなんて
メズラシイよ！
ハナから
まちがっている
というんだものね



●よく出るエラーメッセージ

シンタックス エラー Syntax error

プログラムが文法に合っていないというエラー。

実は、入力するときの打ちまちが多い。

イリーガル ファンクション コール Illegal function call

ステートメントや関数の使い方がまちがっている。指定できる範囲をこえて、指定してしまったなどがある。

アウト オブ データ Out of DATA

READ~DATAで、DATA文に読み出せるデータがない。

リターン ウィズアウト ゴーサブ RETURN without GOSUB

GOSUB文とRETURN文が、うまくかみ合っていない。

タイプ ミスマッチ Type mismatch

READ~DATAなどで、データの形式と変数の形式がまちがっている。

デバイス アイオー エラー Device I/O error

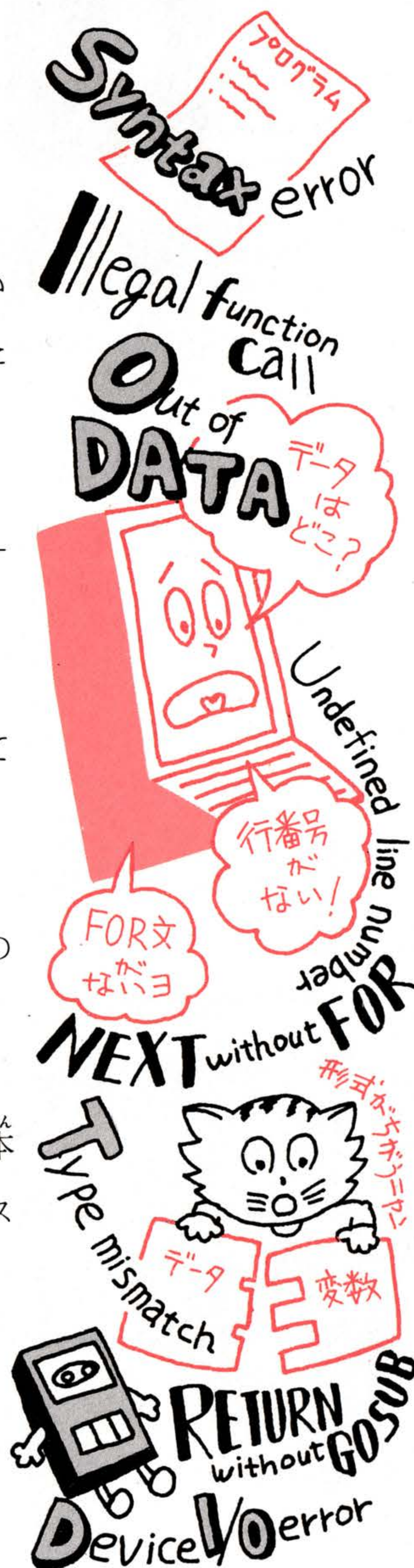
カセットにSAVEしたプログラムが、うまく本体にLOADできない。本体とカセットの接続ミスや音量ミスのことが多い。

ネクスト ウィズアウト フォー NEXT without FOR

NEXT文に対応するFOR文がない。

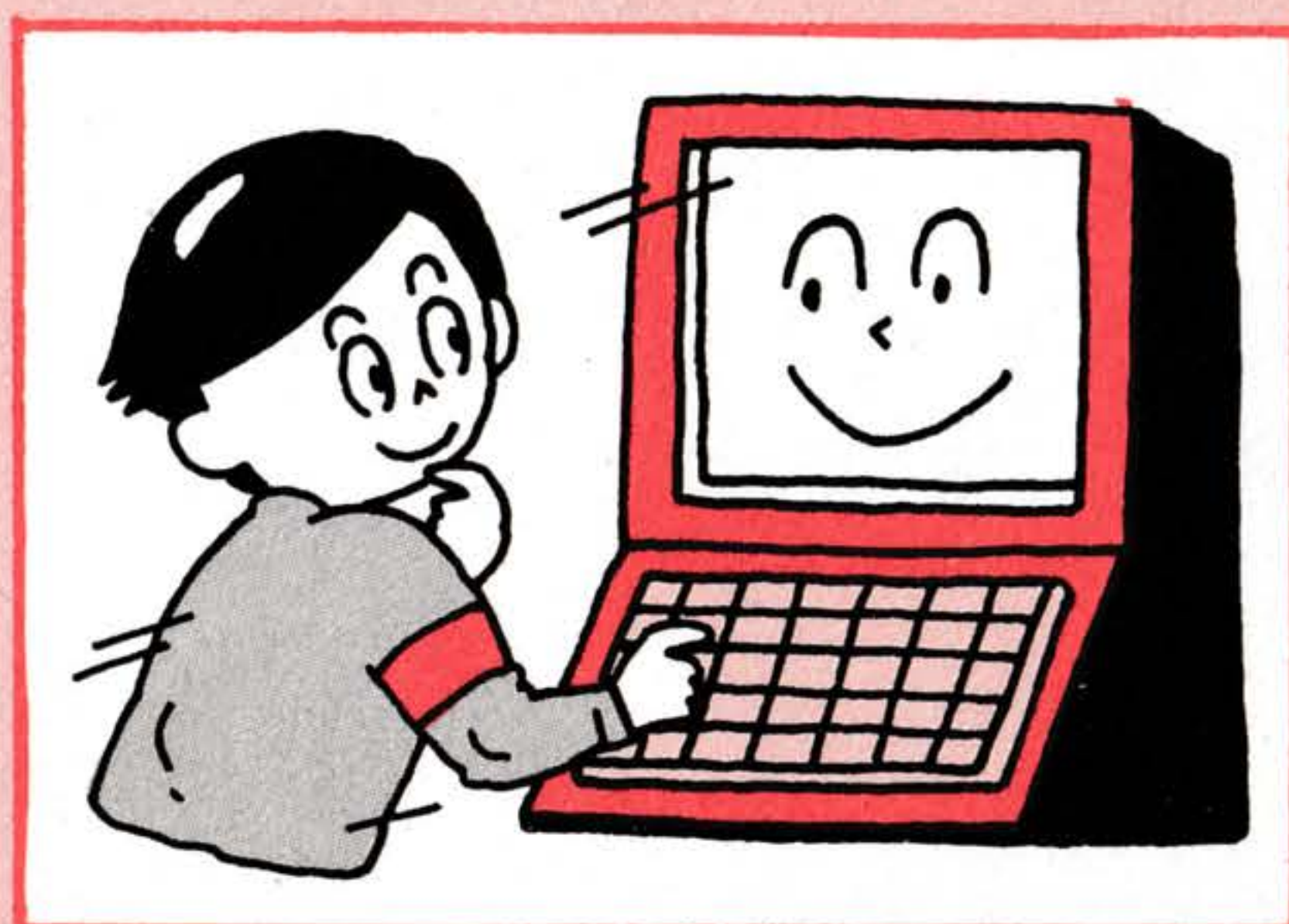
アンデファインド ライン ナンバー Undefined line number

指定した行番号がプログラムにない。



パート3

やさしい プログラミング



3 1

プログラミングとは



めい れい し ごと
命令をつないで仕事をさせること

フローチャート ● ^{し ごと} ^{なが} ^ず ^か **仕事の流れを図に書いてみるとよくわかる**

BASICの^{き ほん めい れい}基本命令をひと^{とお}通りマスターしたら、
次はプログラムの^{つく}作り方を^{かた}勉強^{べん きょう}しよう。

プログラミングとは、コンピュータに^{もくてき}目的の^し仕事を^{ごと}させるために、^{めい れい}命令と^{めい れい}命令をつな^あぎ合^あわせて
プログラムを^{つく}作ることなのだ。

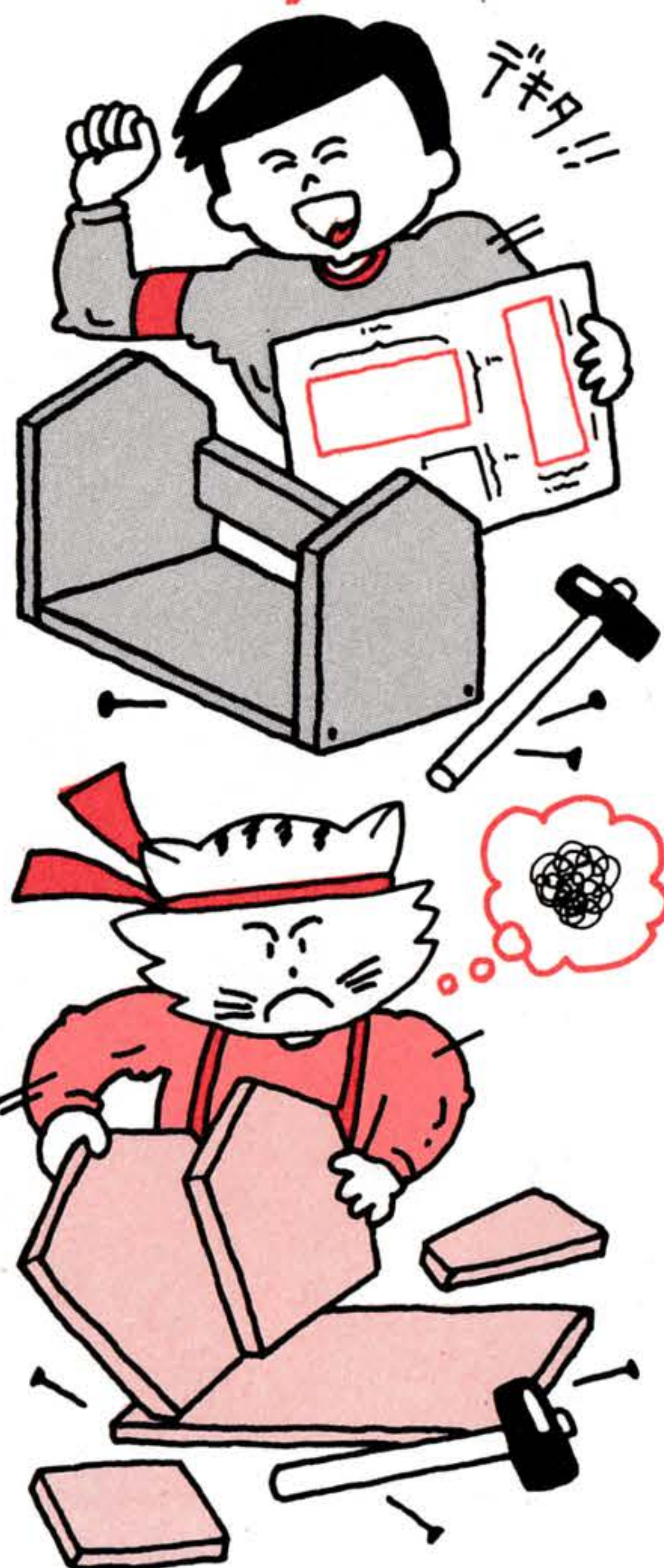
BASIC命令を知^しっていても、それをちゃん^{めい れい}とつ
な^あぎ合^あわせることができなければ、プログラムを
作^{つく}ることはできないんだよ。

プログラムを^{つく}作^{つく}るには、まずどんな^{し ごと}仕事をさせ
たいかを^きはっきり決^きめる。次に、どんな^{じゅん じょ}順序でそ
の^{し ごと}仕事をさせるか、^{じゅん ばん}順番を決^きめてやる。その^{じゅん ばん}順番
ごとに、BASIC命令を^{めい れい}あては^あめるんだ。

といっても、いきなりキーボードからプログラ
ムを^{にゅうりよく}入力するのは、^{しょしんしゃ}初心者の^{きみ}きみにはまだ^{はや}早い
のではないかな？

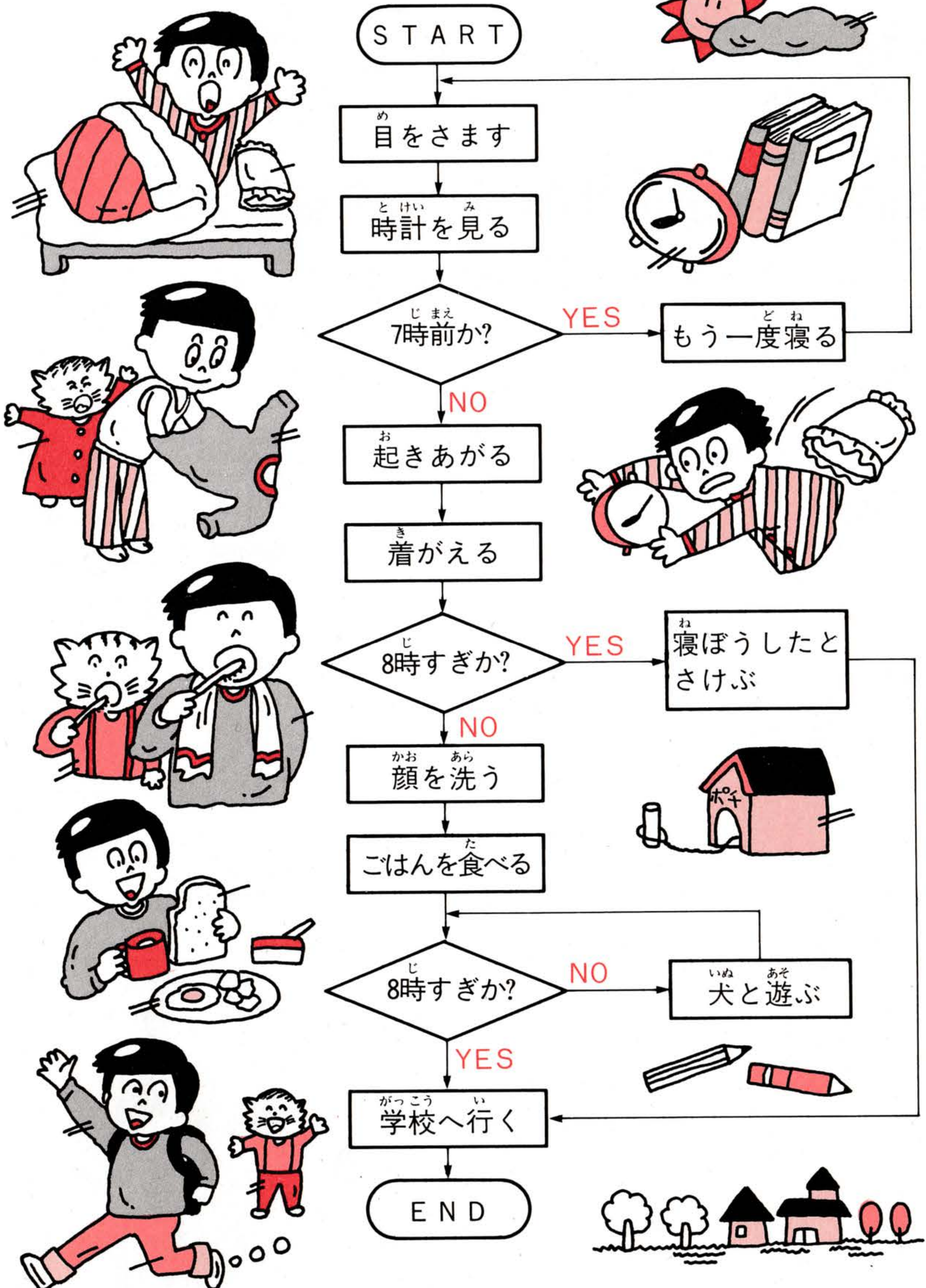
そこで、^{し ごと}仕事の^{なが}流れを^ず図にあらわしてみよう。
この^ず図が「フローチャート」とか「^{なが}流れ^ず図」とよ
ばれるものなんだ。

フローチャートを書
けば、^{ふく ざつ}複雑な^{し ごと}仕事で
も、その^{ないよう}内容がひと
目^めでわかってしまう



●フローチャートの書きかた

あさ お がっこう い
●朝起きて学校へ行くまでのフローチャート





さんかくけい めんせき けいさん 三角形の面積を計算する

インプットめいれい INPUT命令 ● キーボードからデータを入力する

インプットめいれい INPUT命令は、キーを押してデータを入力するときに使う。

①
10 CLS
20 INPUT A
30 INPUT B
40 PRINT A*B

②
10 CLS
20 INPUT "A=" ; A
30 INPUT "B=" ; B
40 PRINT A*B

まず左のプログラムをRUNさせてごらん。画面は“?” (クエスションマーク) を出してとまったままだね。このままなにもしなければ、永遠に止まったままだ。

そこで、1でも2でもいいから、きみの好きな数字を入力してみよう。この数字が変数Aに代入される。

また?が出たね。もうひとつ数字をキー入力しよう。この数字が変数Bに代入される。

それから、40行の仕事、 $A \times B$ のかけ算をしてその答えを画面に表示してくれる。

右のプログラムをRUNさせてみよう。こんどは?だけかわりに「A=?」、「B=?」と画面に表示される。つまりメッセージつきだから、意味がわかりやすいというわけだ。

AとBに入りたい数字をキーボードから入力すると、同じように計算して答えを出してくれるよ。

つぎ
次の2つのプログラムを入力しよう

どちらのプログラムも、同

じ仕事をするんだよ

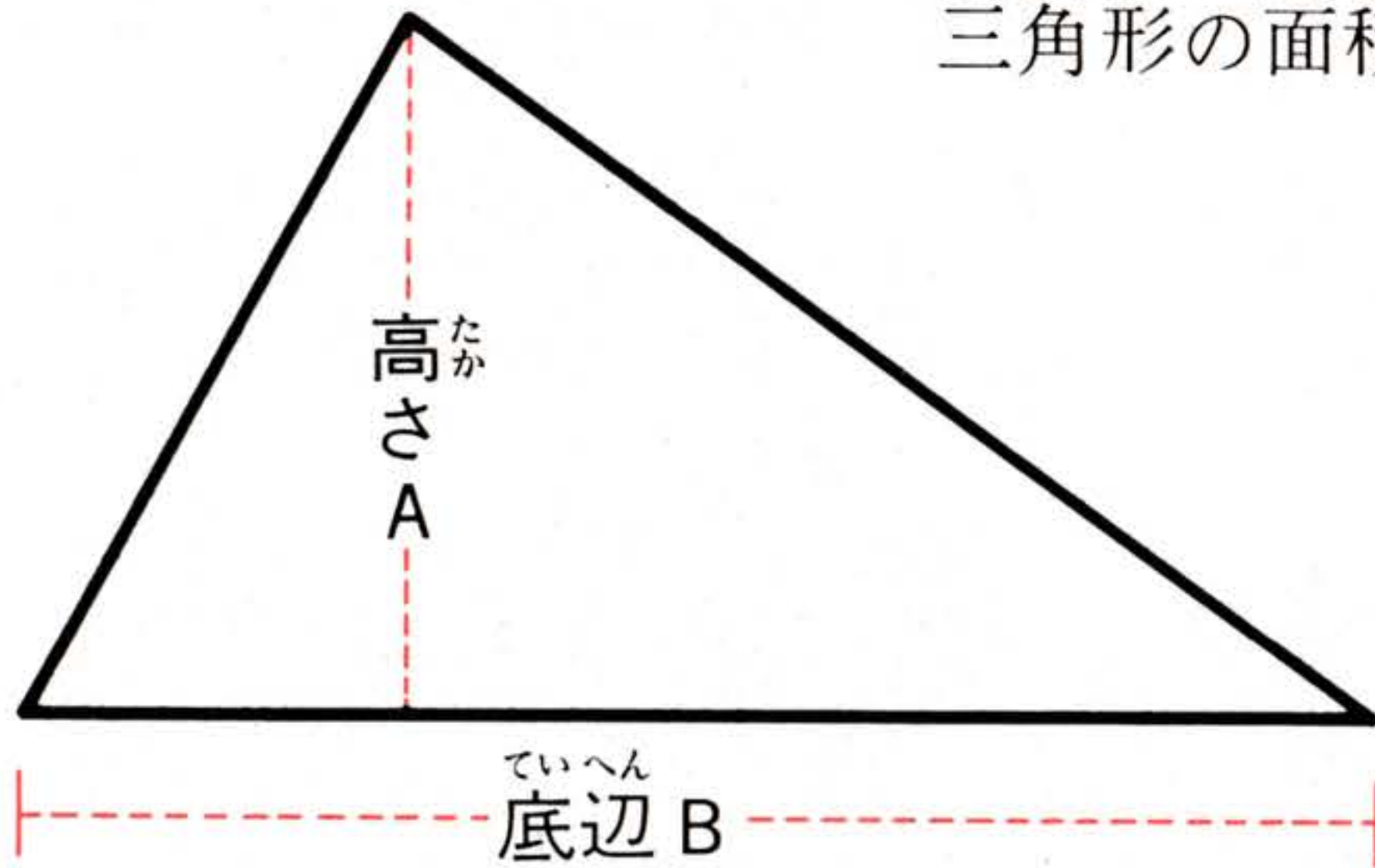


メッセージつきの
インプットめいれい
INPUT命令のときは、
変数の前に; (セミ
コロン) をつけ忘れないようにね



●三角形の面積を計算するフローチャート

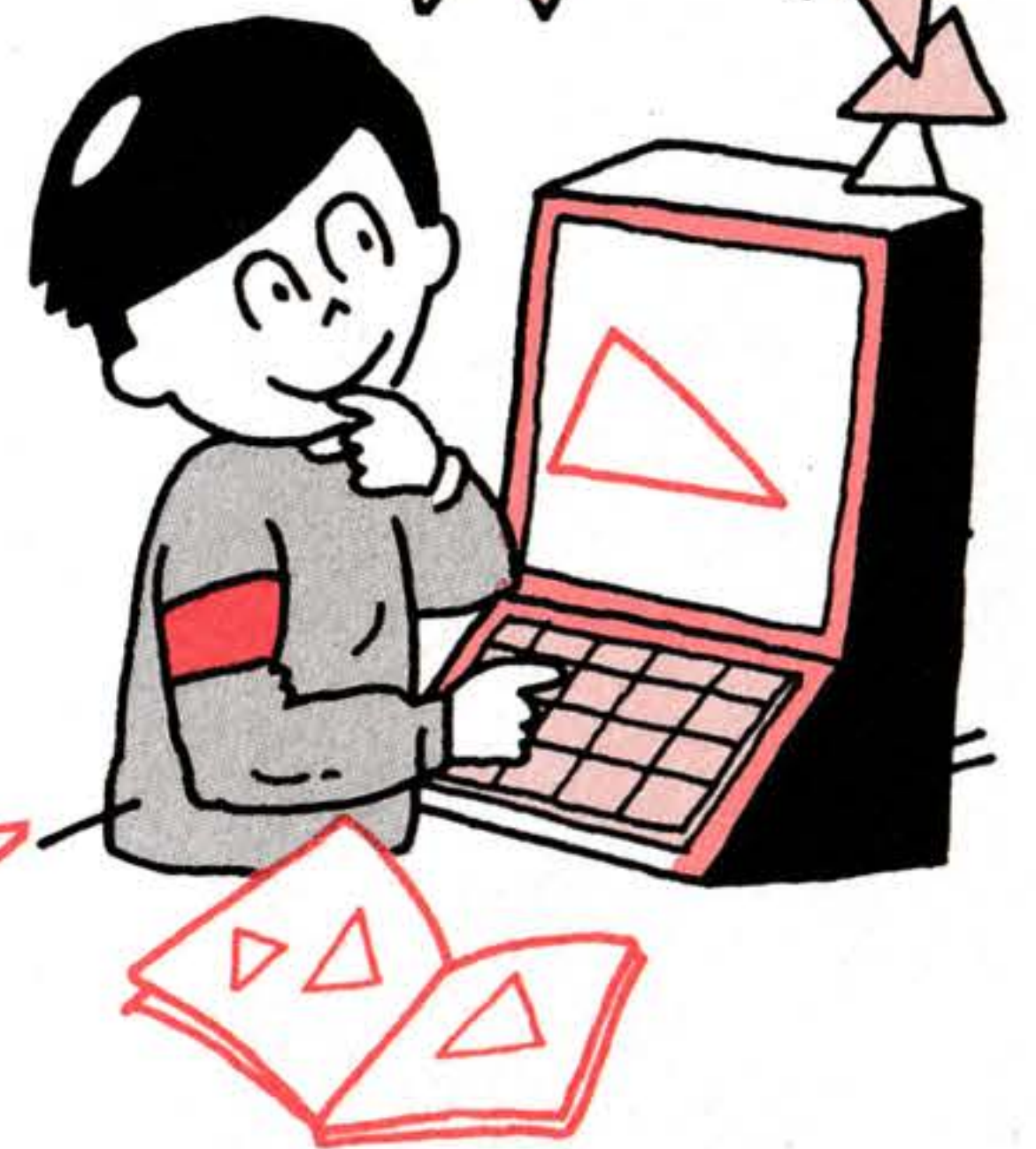
さんかくけい めんせき けいさん
三角形の面積をコンピュータに計算させよう。



さんかくけい めんせき けいさん
三角形の面積の計算は、「底辺×高さ÷2」だね。でも、このままだをコンピュータにいったってわからないよ。

コンピュータに仕事をさせる順序でプログラミングしてやらなければならない。

コンピュータの仕事は、「キーボードから三角形の底辺の長さと高さを入力してやると、その面積を計算して画面に表示する」ということだな



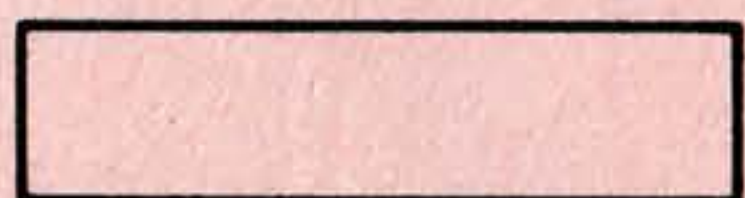
フローチャートに使う記号

フローチャートを書くにも記号や規則があるけれど、いちばんたいせつなのは、ひと目見て流れがわかるということだ。あまり規則にしばられず、どんどん書いていこう。

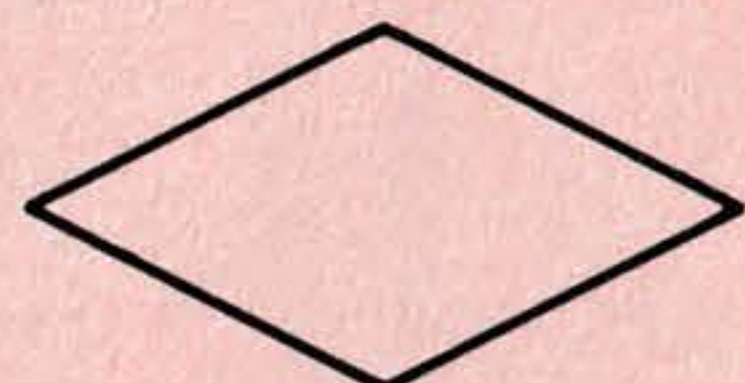
START

END

.....はじめと終わりの記号



.....操作や仕事の内容を書き込む



.....処理内容を分けるための条件を書き込む

では、コンピュータの仕事^{しごと}をフローチャートにしてみよう。

ヒントだよ

INPUT命令^{めいれい}.....

INPUT命令^{めいれい}.....

計算式^{けいさんしき}.....

PRINT命令^{めいれい}.....

スタートマークと
エンドマークは忘
れずに



START

底辺^{ていへん}を入力^{にゅうりよく}

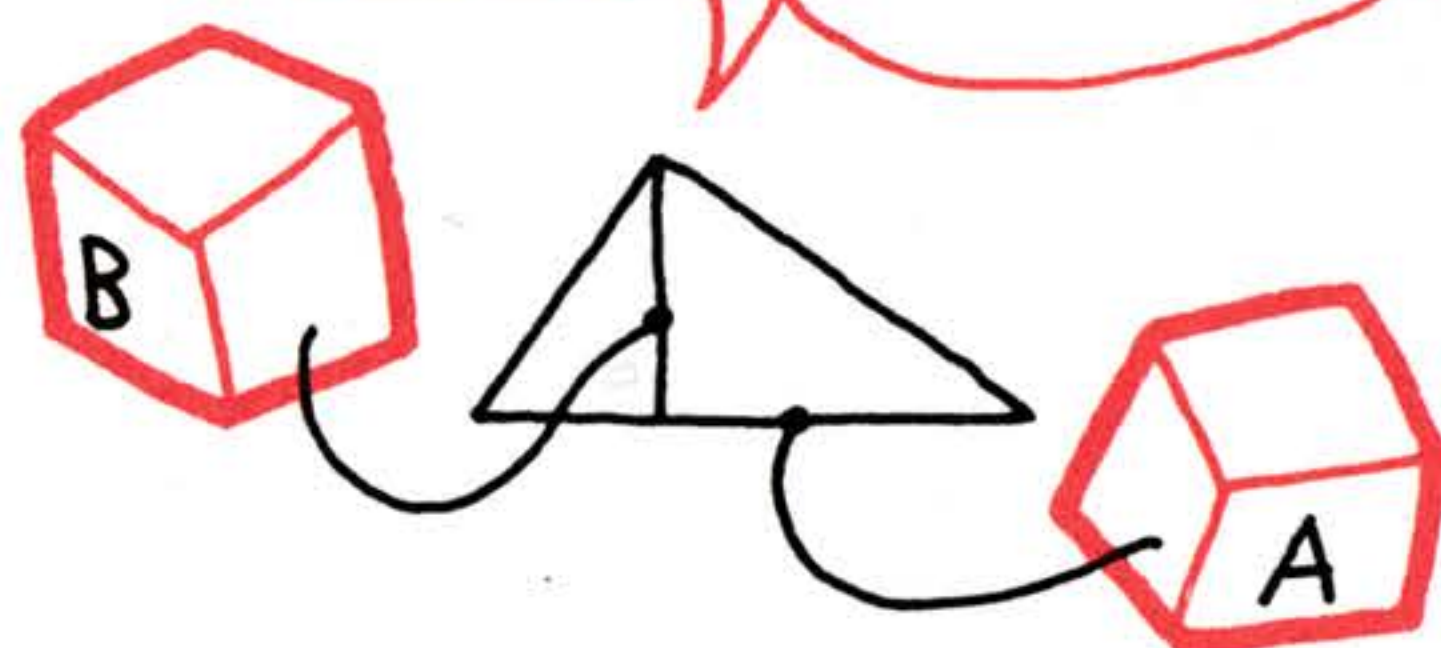
高さ^{たか}を入力^{にゅうりよく}

面積^{めんせき}計算^{けいさん}

結果^{けっか}を表示^{ひょうじ}

END

三角形^{さんかくけい}の底辺^{ていへん}の長さ^{なが}を変^{へん}数^{すう}A、高さ^{たか}を変^{へん}数^{すう}B、面^{めん}積^{せき}を変^{へん}数^{すう}Cにして、プロ
グラミングしてごらん。
むずかしいプログラムじ
ゃないから、きみにもで
きるはずだ



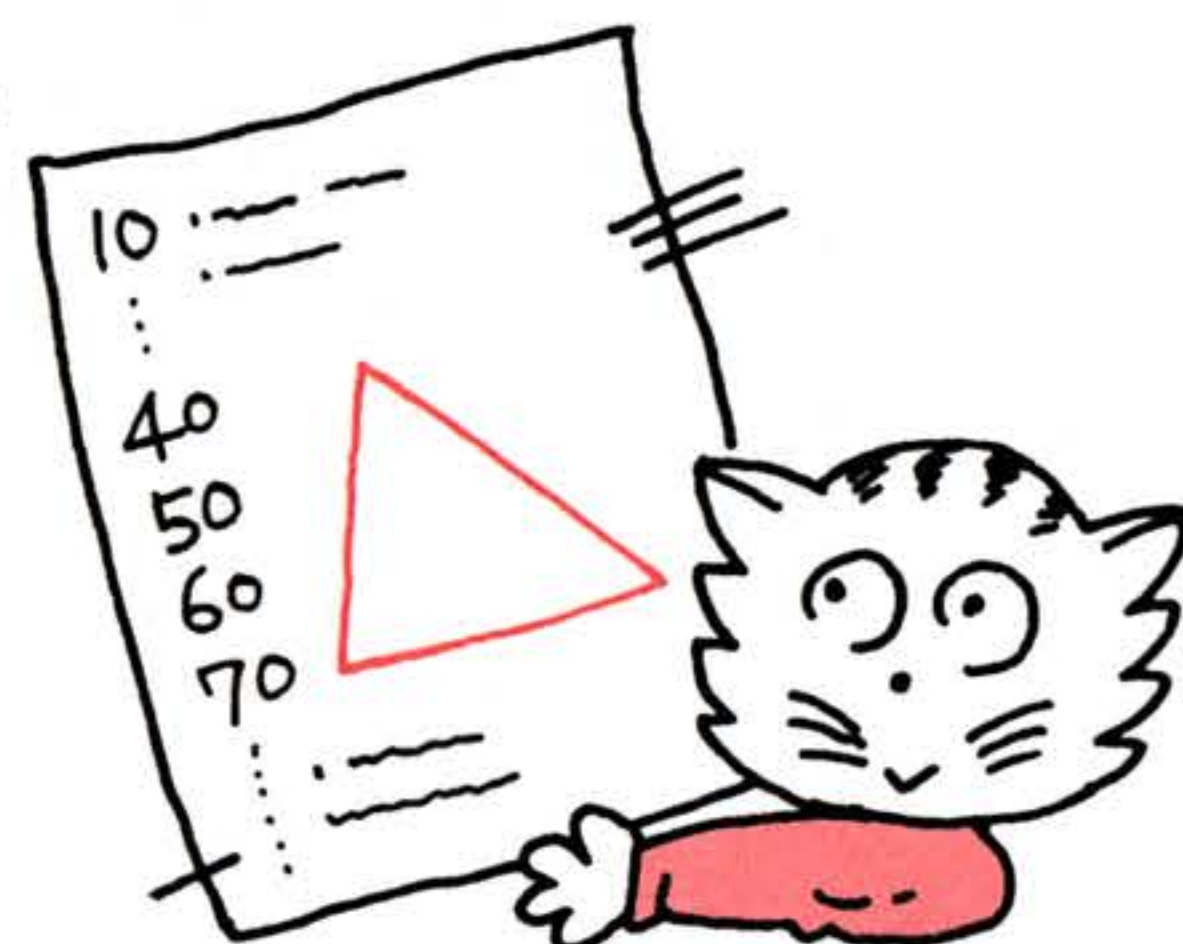
フローチャートにBASIC命令^{めいれい}をあては
めてみよう。

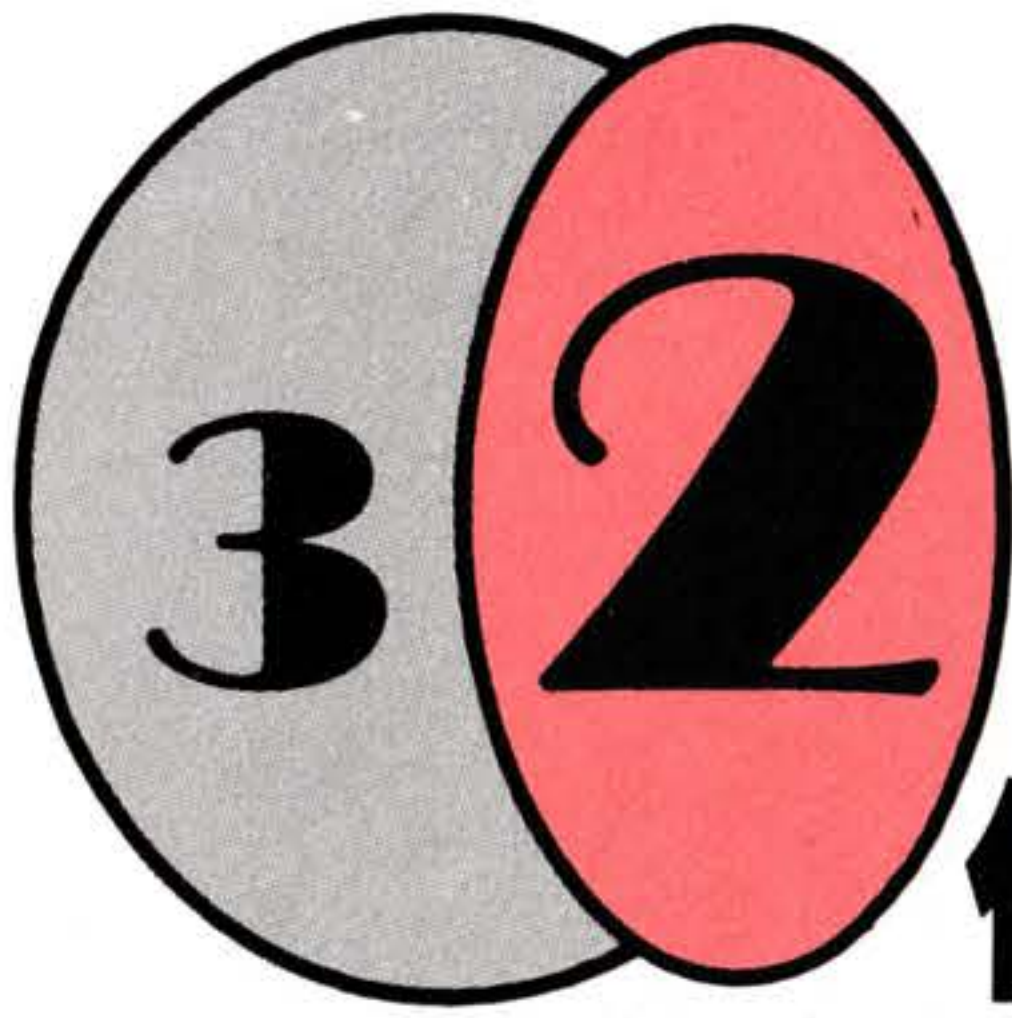
```
10 CLS
20 LOCATE 5,3
30 PRINT"さんかくけい / めんせき"
40 INPUT"テイヘン=" ; A
50 INPUT"タカサ =" ; B
60 C=A*B/2
70 PRINT"めんせき は " ; C
```

底辺^{ていへん}の長さ^{なが}を入力^{にゅうりよく}
高さ^{たか}を入力^{にゅうりよく}する
三角形^{さんかくけい}の面積^{めんせき}を計算^{けいさん}する
結果^{けっか}を表示^{ひょうじ}する

三角形^{さんかくけい}の面積^{めんせき}を計算^{けいさん}するプログラムは、40行^{ぎょう}か
ら70行^{ぎょう}までだということは、わかるね。

ローケート^{ローケート}命令^{めいれい}とPRINT命令^{プリントめいれい}で、何^{なん}のプログラ
ムかわかるように、タイトルをつけておいたよ。
これで完^{かん}ぺきだね。





仕事の流れが違うとき



IF~THENのフローチャート

きみがキーボードから入力した^{にゅうりょく}数が、ぐう^{すう}数かき^{すう}数かをコンピュータが^{くべつ}区別するプログラムを作ってみよう。

コンピュータの^{しごと}仕事を大きく^{おお}分けてみると、

1. きみがキーボードから^{かず}数を入力する^{にゅうりょく}
2. ^{すう}ぐう数か、^{すう}き数かを^{くべつ}区別する
3. ^{すう}ぐう数のときには「グウスウ」、^{すう}き数のときには「キスウ」と^{がめんひょうじ}画面表示する

これをフローチャートに^か書いてみるわけだが、その^{まえ}前に、コンピュータがどうやって^{すう}ぐう数とき^{すう}数を^{くべつ}区別するの^{せつめい}かを説明しておこう。

● ^{すう}ぐう数ってどんな^{かず}数？

2、4、6、8、10、12……これらはみんな^{すう}ぐう数だね。3、5、7、9、11……は^{すう}き数だ。

では、^{すう}ぐう数とはどんな^{かず}数をいうのかな？ ^{かんが}考
えてみよう。

^{すう}ぐう数とは、2で^わ割り切れる^き数だといえないか
な？ ^{ぎやく}逆に、^{すう}き数は2では^わ割り切れず、^{あま}余りが出
る^{かず}数だね。

グウスウ

キーボード
から、2を
^{にゅうりょく}入力すると



キスウ

こんどは5
^{にゅうりょく}を入力だ

アレッ？

どうしてわかるのか
なあ



コンピュータがぐう数かき数かを区別するのに
 使う記号がMODだ。MODはわり算をしたときの
 余りの数を出すときの記号なんだ。だから、

$$A \text{ MOD } 2 = 0$$

という式だったら、Aの数を2で割ったときの余
 りは0ということだ。余りが0なら、Aはぐう数
 だね。余りが1なら、Aはき数だ。

これがわかれば、プログラムは作れるよ。さあ、
 フローチャートを書いてみよう。

Aに6を入れてみる

$$6 \text{ MOD } 2$$

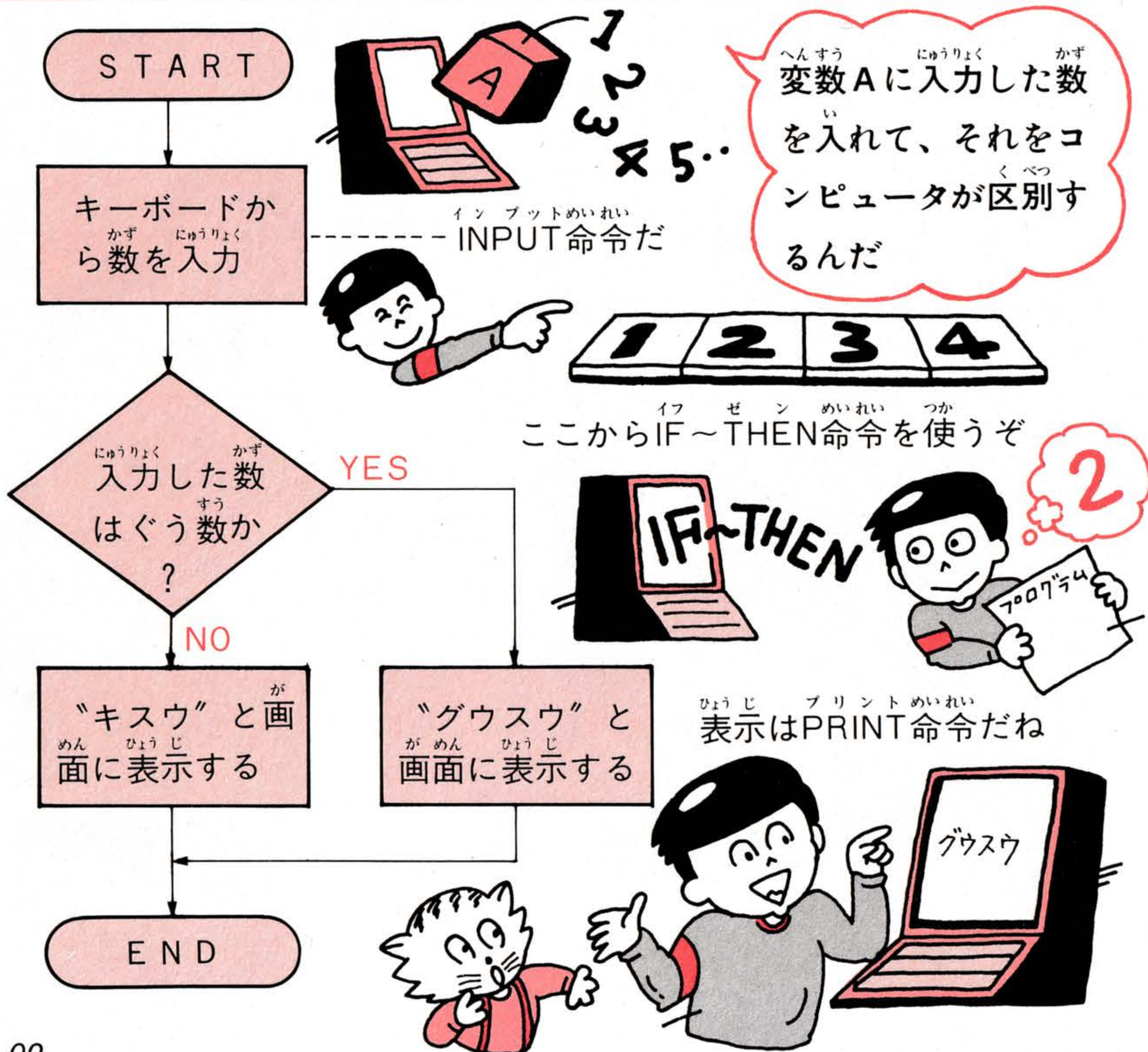
6 ÷ 2は、わり切れ
 るから余りの数は

$$6 \text{ MOD } 2 = 0$$

だからぐう数だね



●ぐう数か、き数かを区別するフローチャート




```
10 INPUT A
20 IF A MOD 2=0 THEN PRINT"グウスウ"
:ELSE PRINT"キスウ"
30 END
```

①

```
10 INPUT A
20 IF A MOD 2=0 THEN 50
30 PRINT"キスウ"
40 GOTO 60
50 PRINT"グウスウ"
60 END
```

②


プログラムは2通り作れるよ。20行がポイントだ。



①もしAに代入した数がわり切れる数なら“グウスウ”と表示しなさい。

②もしAに代入した数がわり切れる数なら50行へ行きなさい。NOなら30行だ。

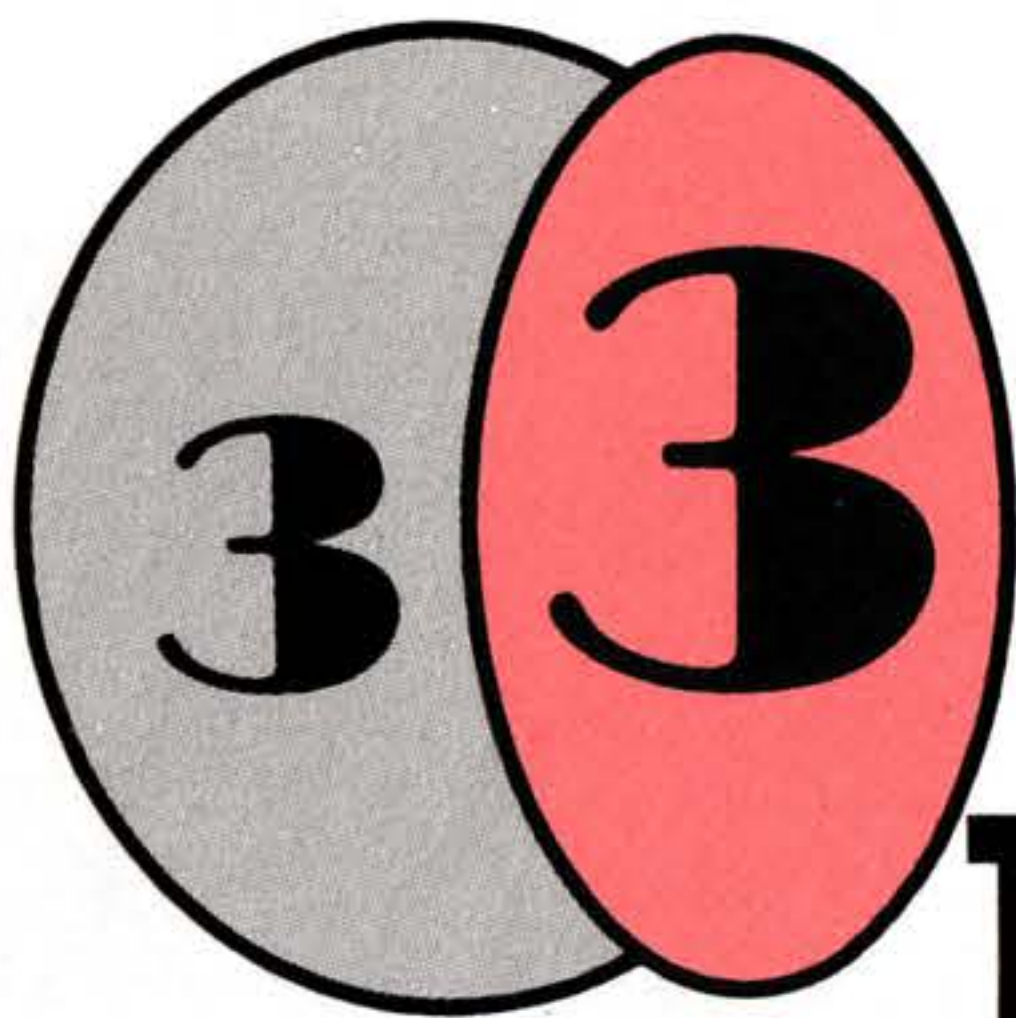


IF～THEN～ELSE命令は  (ひし形) で囲む

IF～THEN～ELSE命令は、いろいろな条件によってコンピュータが処理する仕事の内容が違うときに使う命令だ。

どの条件のときにどんな仕事を実行するか、条件が合わないときはどういう仕事をするかを、それぞれ分けて書いてやらなければならない。

ひし形の囲みの中には？マークを書いておくこと。また、条件が合ったときは(YES)、合わなかったときは(NO)と書いて、それぞれの仕事の行き先を矢印で結ぶ。



1から100までのたし算



フオー ネクスト イフ ゼン FOR~NEXTがIF~THENで

1から100までの数をたし算するといくつになるかという問題だ。1 + 2 + 3 + 4 + 5 + 98 + 99 + 100 = ? きみが計算すると何分かかるかな? コンピュータならアッというまだぞ。ただし、計算するためのプログラムを作らないとね。

このたし算の考え方を説明しておこう。

2は1より1多い数。3は2より1多い数。4は3より1多い数だね。つまり、前の数よりも1多い数を順番にたしていけばいいわけだ。

そこでまず、A、Bという箱を2つ用意しよう。

Aの箱には次にたす数を入れる。Bの箱はたし算の結果を入れる箱としよう。

この箱は変数だから、新しい数を入れると前の数は出てしまう性格があったね。これを利用しようというわけだ。どんどん入ったり出たりして、Bに100までたした数が入ったら計算はおわり。

このプログラムは、IF~THEN命令を使っても作れるし、FOR~NEXT命令でも作れるから、2通りの作り方を説明しておこう。

$$1 + 2 = 3$$

$$3 + 3 = 6$$

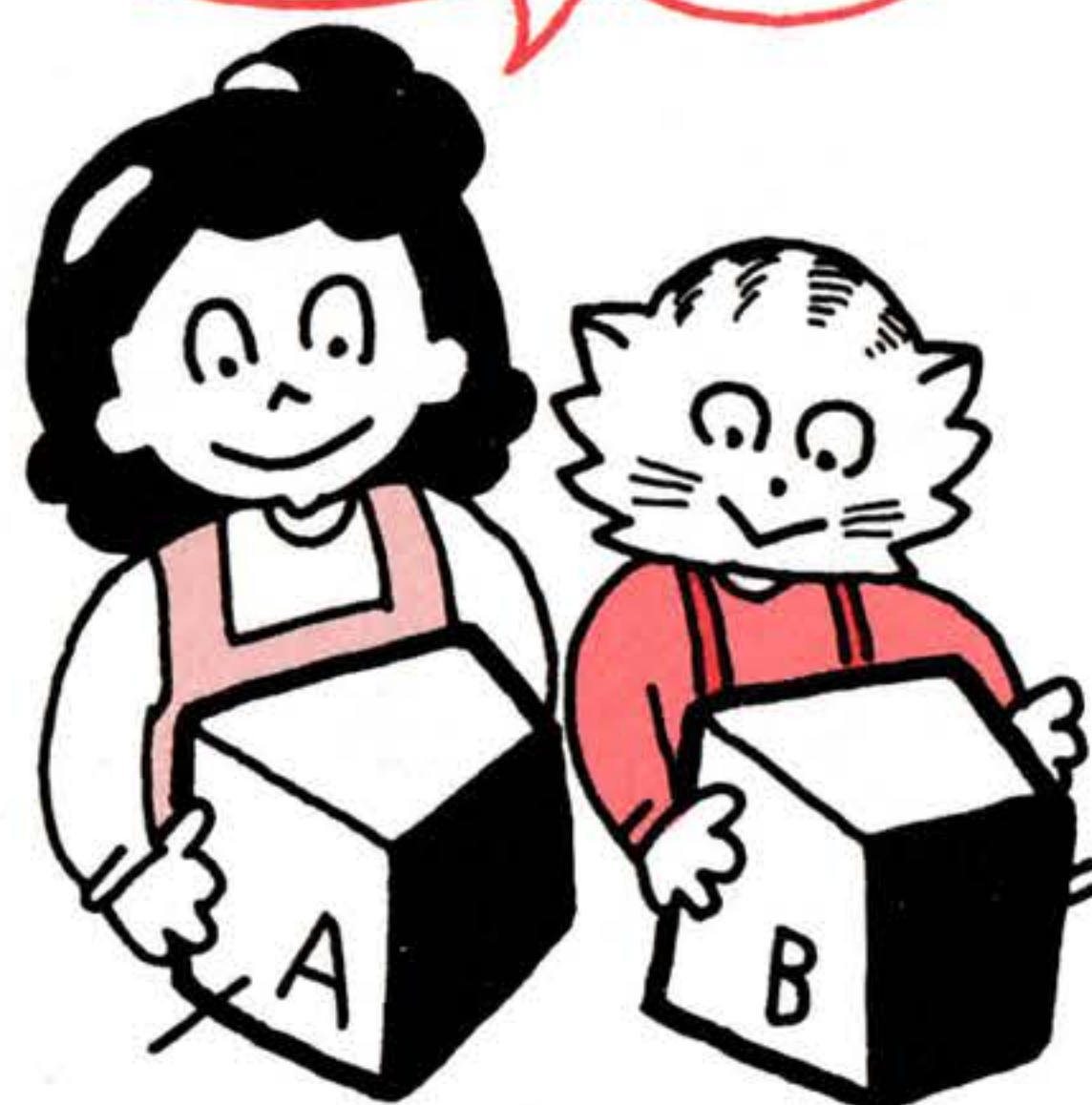
$$6 + 4 = 10$$

フウ〜。

たいへんだなあ



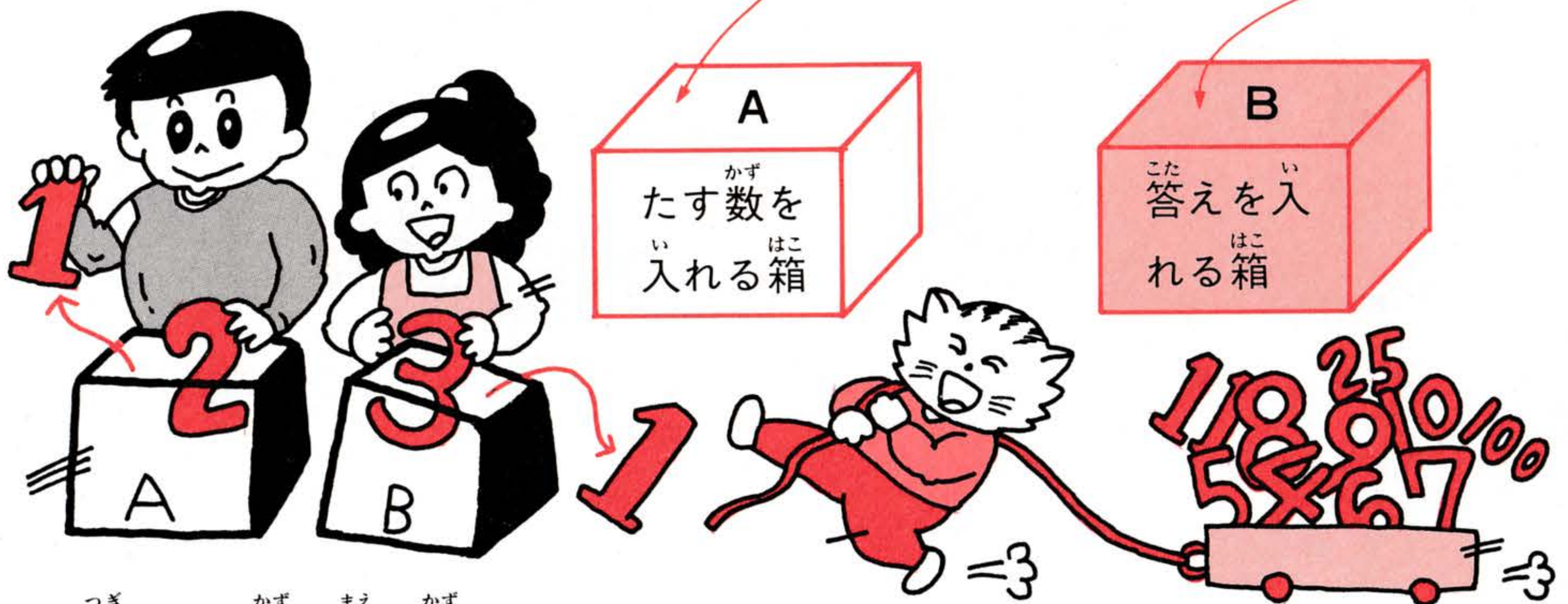
次のページに進む前に、自分で紙に箱を書いて考えてごらん



●変数を使って1から100までのたし算をする

A、B 2つの箱を用意する

はじめは1が入っている だからBも1



つぎ 次にたす数は前の数より

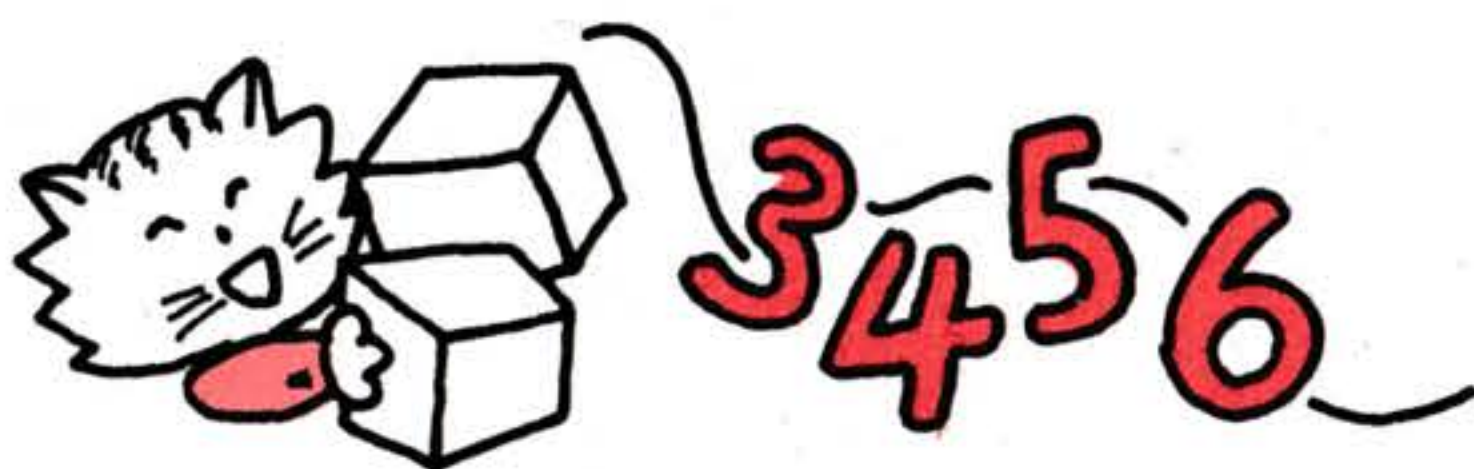
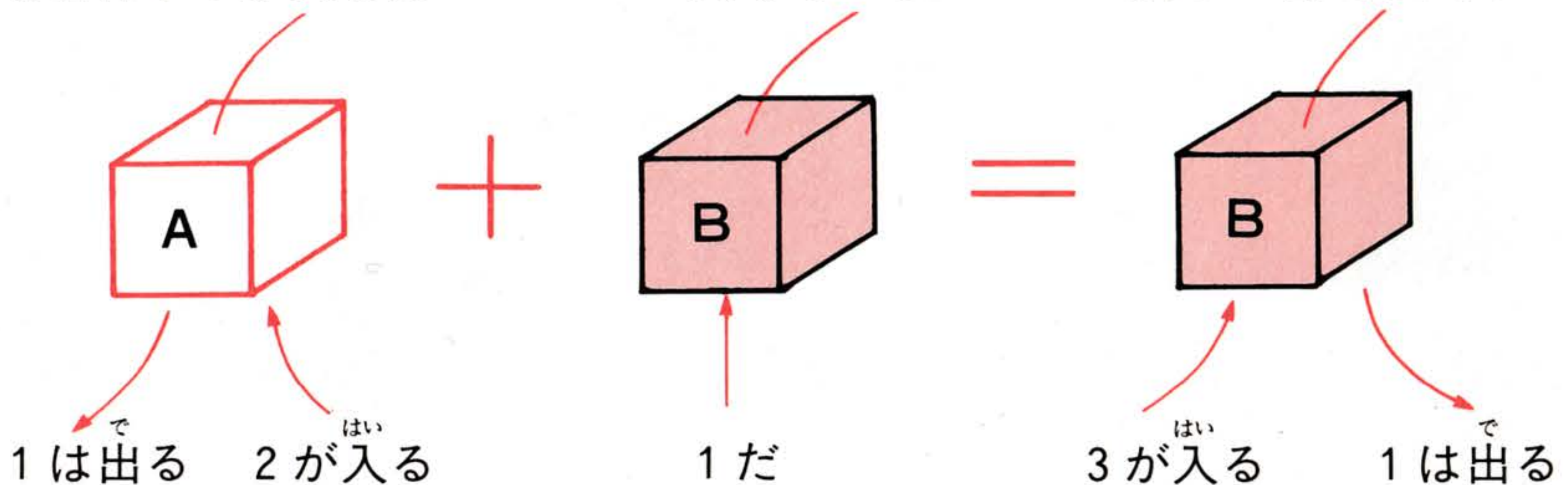
1多い数が入る。式にす

ると $A + 1$ が入るよ

さいしょ 最初の答えが

入っている

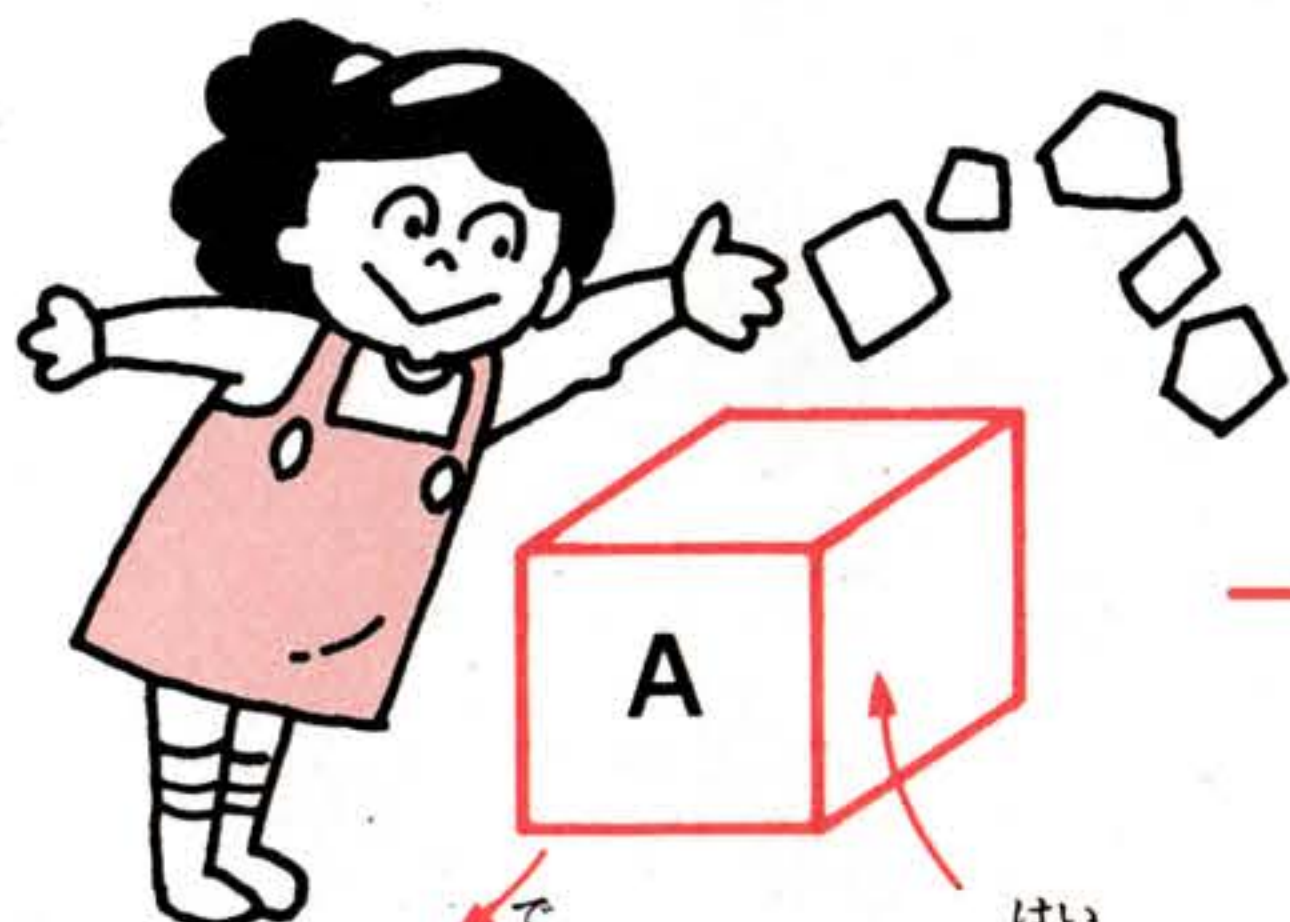
あたら 新しい答えが入る



まえ 前のAより1多い数 ($A + 1$)

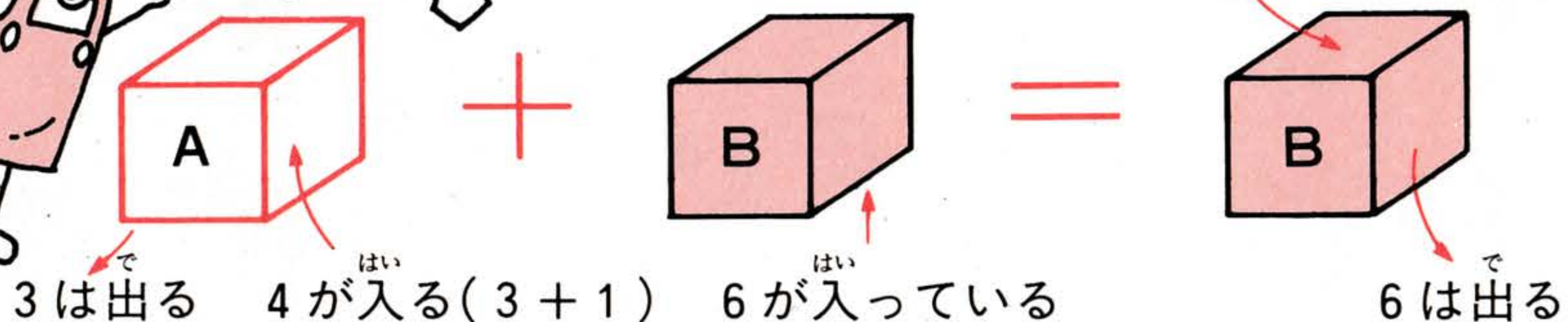
もうひとつやってみよう

$1 + 2 + 3 + 4$ までの答え



Bに入っている答え 6

あたら 新しい答え10が入る



Aの箱に100が入るまで
たし算を続けるよ



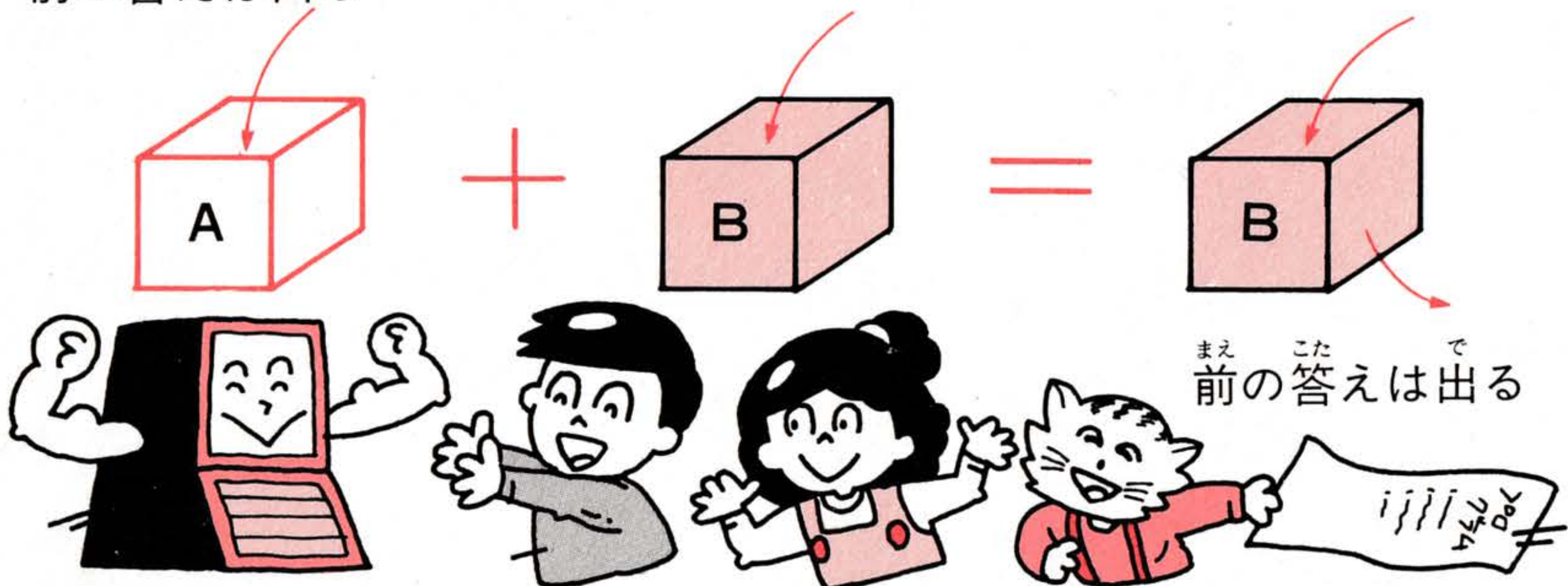
= (イコール) は代入する記号
だったことを思い出してね



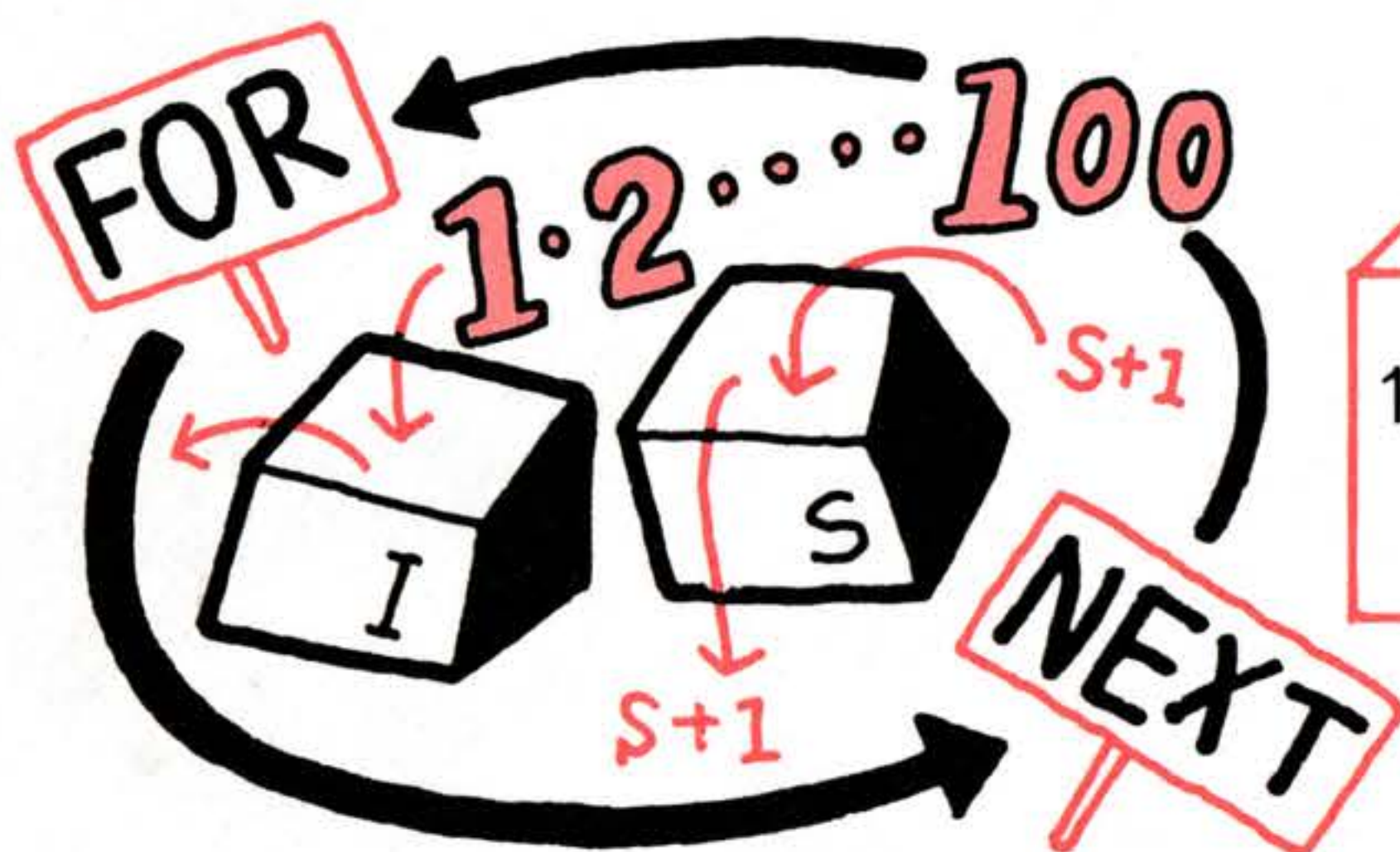
A + 1 が100になった
99 + 1 ということだね
前の答えは出る

1 から99までの数
をたした答え

1 から100までたし
た答えが入った

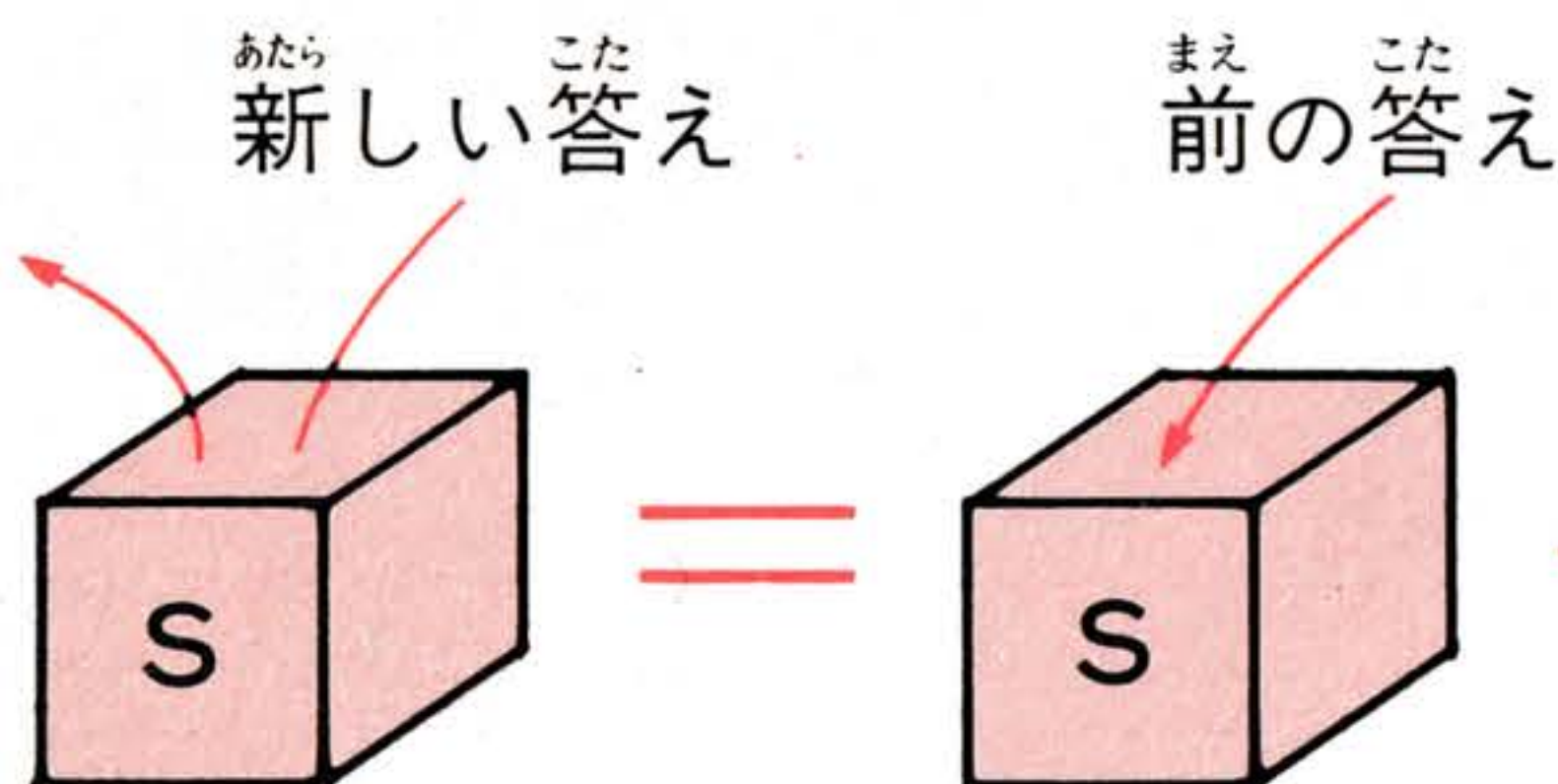


FOR~NEXTなら1から100までをどんどん入れる



I
1 から100を
入れる箱

S
こた
えを入
れる箱



I = 1 TO 100 とすれば

1 から100までどんどん入

っていく。い
ちいち1をた
さなくていい。

さあ、フローチャートだ！



● IF～THEN命令を使ったプログラム

```

10 A=1
20 B=B+A
30 IF A<100 THEN A=A+1:GOTO 20
40 PRINT B
50 END
    
```

①

A < 100 の意味は、A が 100 より小さい数ということさ

● FOR～NEXTを使ったプログラム

```

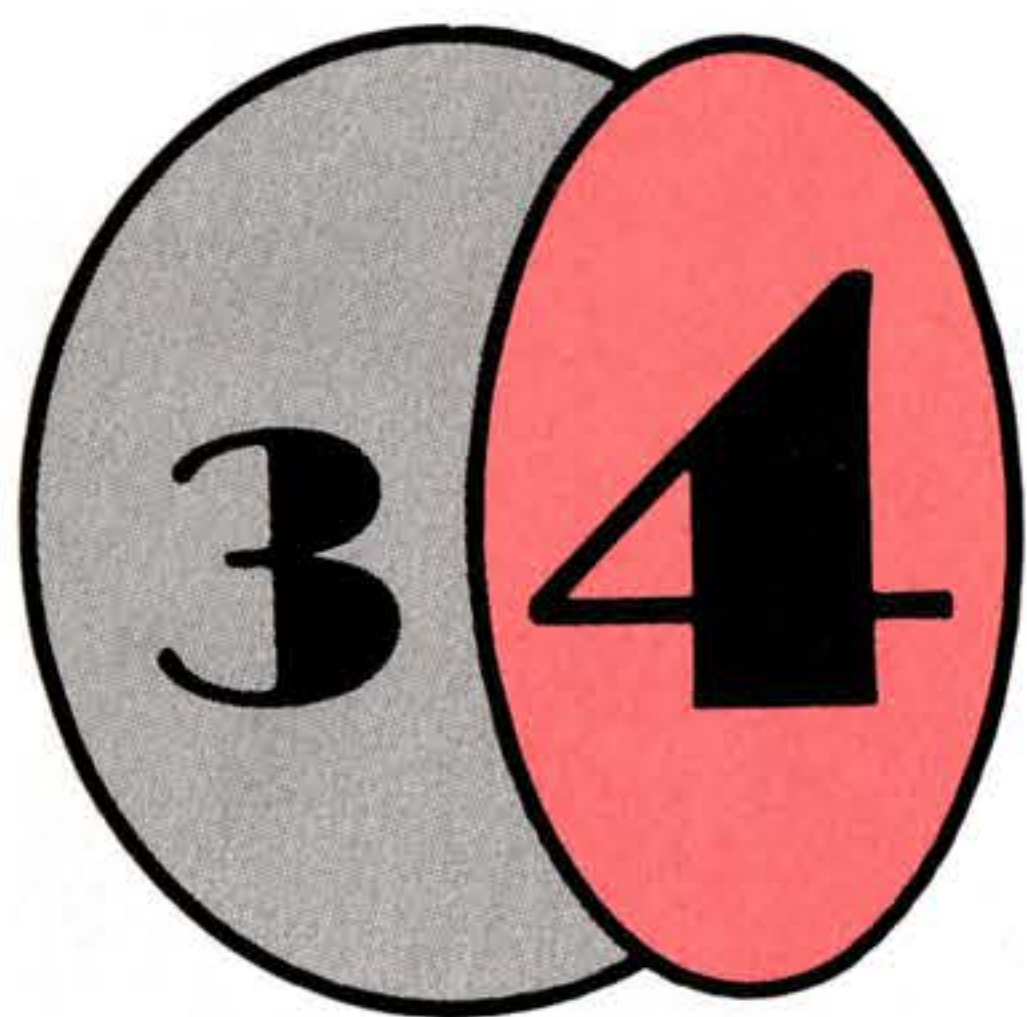
10 FOR I=1 TO 100
20 S=S+I
30 NEXT I
40 PRINT S
50 END
    
```

②

FOR～NEXT

だと、ずいぶんかんたんになるなあ





サイコロころころ



パソコンに^{かず}数^{つく}を作らせる

^{ランダム}かんすう ^{インテジャー}かんすう **RND関数とINT関数** ● 2つそろわなければ^{せいすう}整数は作れない

キーボードから^{かず}数^{にゅうりよく}を入力するときは、1でも3でも10でも、^す好きな^{かず}数^{にゅうりよく}を入力することができるね。

では、コンピュータに^{かず}数^だを出させるにはどうしたらいいのだろう。

コンピュータに^{かず}数^{つく}を作らせるのが、^{ランダム}かんすう RND関数だ。
^{かんすう}関数などという^{なまえ}名前はおぼえなくてもいいけれど、
^{つき}次の^{きほん}基本だけはおぼえておこう。

RND (1)

これは、コンピュータに^{かず}でたらめな^{はっせい}数^{すこ}を発生させる^{めいれい}命令だ。

^{つき}次のプログラムを^{じっこう}実行してごらん。

10 CLS

20 PRINT RND (1)

30 GOTO 20

^{がめん}画面には、^{すうじ}どんどん数字がならんでいくぞ。この^{かず}数はまったくでたらめに^{つく}作られているんだ。

^{ランダム}かんすう RND関数というのはでたらめの^{かず}数^{つく}を作る^{はたら}働きをするんだよ。

^{かんすう}関数とは、こちらから^{あた}データを与えると、^{たい}それに対して^{けいさん}計算や^{そうさ}操作を行なって、^{おこ}その^{けつ}結果を^{かえ}返してくれるものだ

^{かんすう}関数には^{べんり}便利なものがたくさんある。^{すこ}少しずつ^{べんきょう}勉強していくといい



● 0以上1未満の数を生ぜさせてから

0.1398762.....



PRINT RND (1)

ランダム
RND (1) は、0 より
おお
り大きくて1 より小
かず
さい数をでたらめに発生
はっせい
させるんだ。PRINT 命
プリントめい
れい
令をつけて、画面に表
がめん ひょう
じ
示させてみよう。

0 より^{おお}大きくて1 より^{ちい}小さい^{かず}数というのはどん
な^{かず}数だ? そう、小数点付きの0.12345
という^{かず}数だね。

0 より^{おお}大きい^{ちい}のだから 0.0000001.....でもいい。
また、1 より^{ちい}小さくて1 に^{ちか}近い^{かず}数は、0.99999.....
だ。RND (1) は、この0.△×○△××.....とい
う^{かず}数をでたらめに発生^{はっせい}させているが、これより^{おお}大
き^{かず}な^{はっせい}数を生ぜさせるには、どうしたらいいか?

● ^{おお}大きく^{かず}したい^{かず}数だけかける

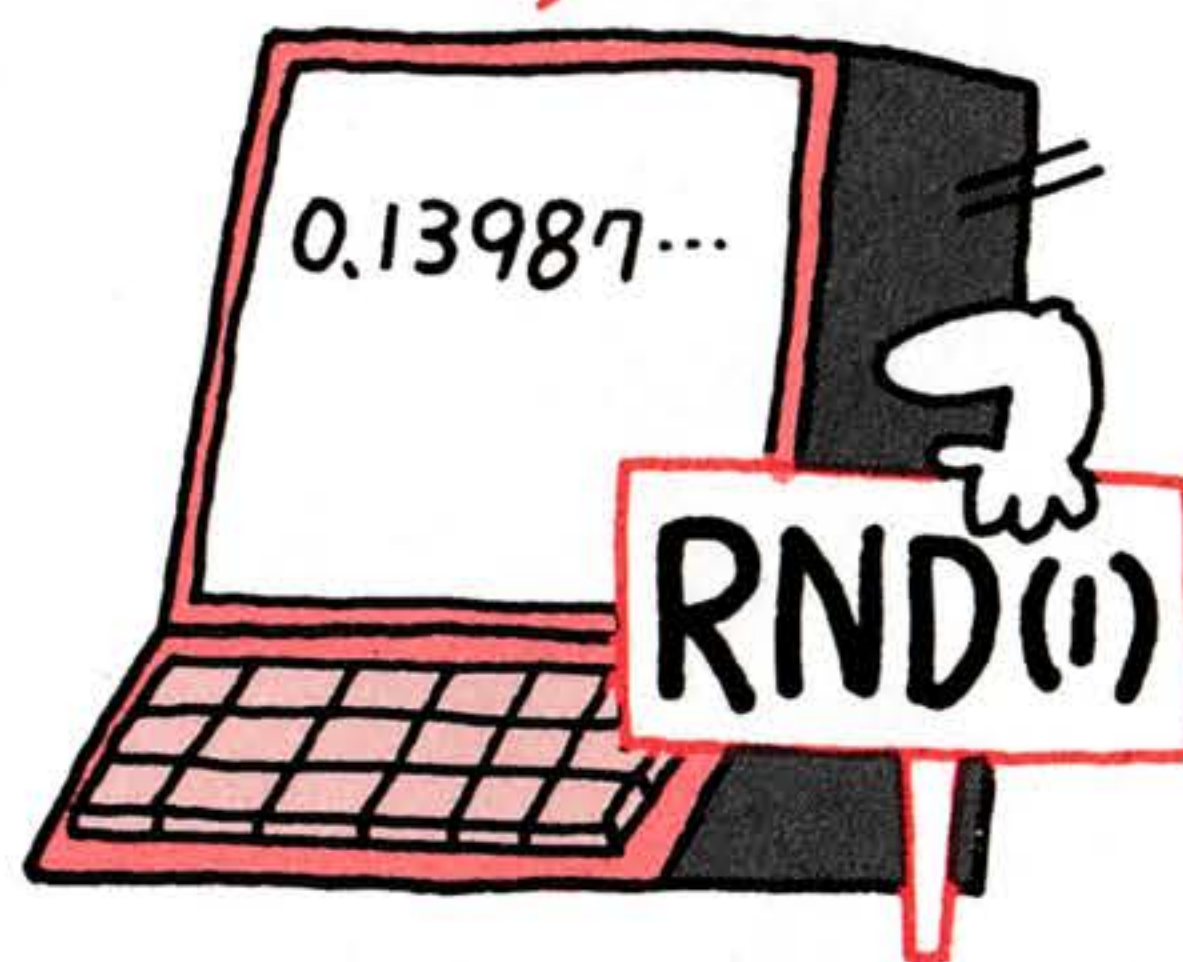
1 よりも^{おお}大きな^{かず}数を作^{つく}りたければ、その^{かず}数だけ
かければいいよ。

たとえば、0 より^{おお}大きくて10 より^{ちい}小さい^{かず}数なら、

RND (1) * 10

これで、0 より^{おお}大きくて10 より^{ちい}小さい^{かず}数がどん
どんできてくる。PRINT^{プリント}で^{ひょうじ}表示してごらん。

このでたらめに発生^{はっせい}
する^{かず}数を、コンピュ
ータでは「乱数^{らんすう}」と
いうんだ



●小数点以下はとってしまう

●小数点付きの数を整数になおすINT文

RNDで数字のもとにはできた。でもまだ小数点以下の数がゴチャゴチャとついたままだ。これをとってしまえば、整数ができるね。

小数点以下をとってしまう命令が、INT文だ。

書き方は、こうなる。

```
INT (RND (1) *10)
```

画面に表示させるには、

```
PRINT INT (RND (1) *10)
```

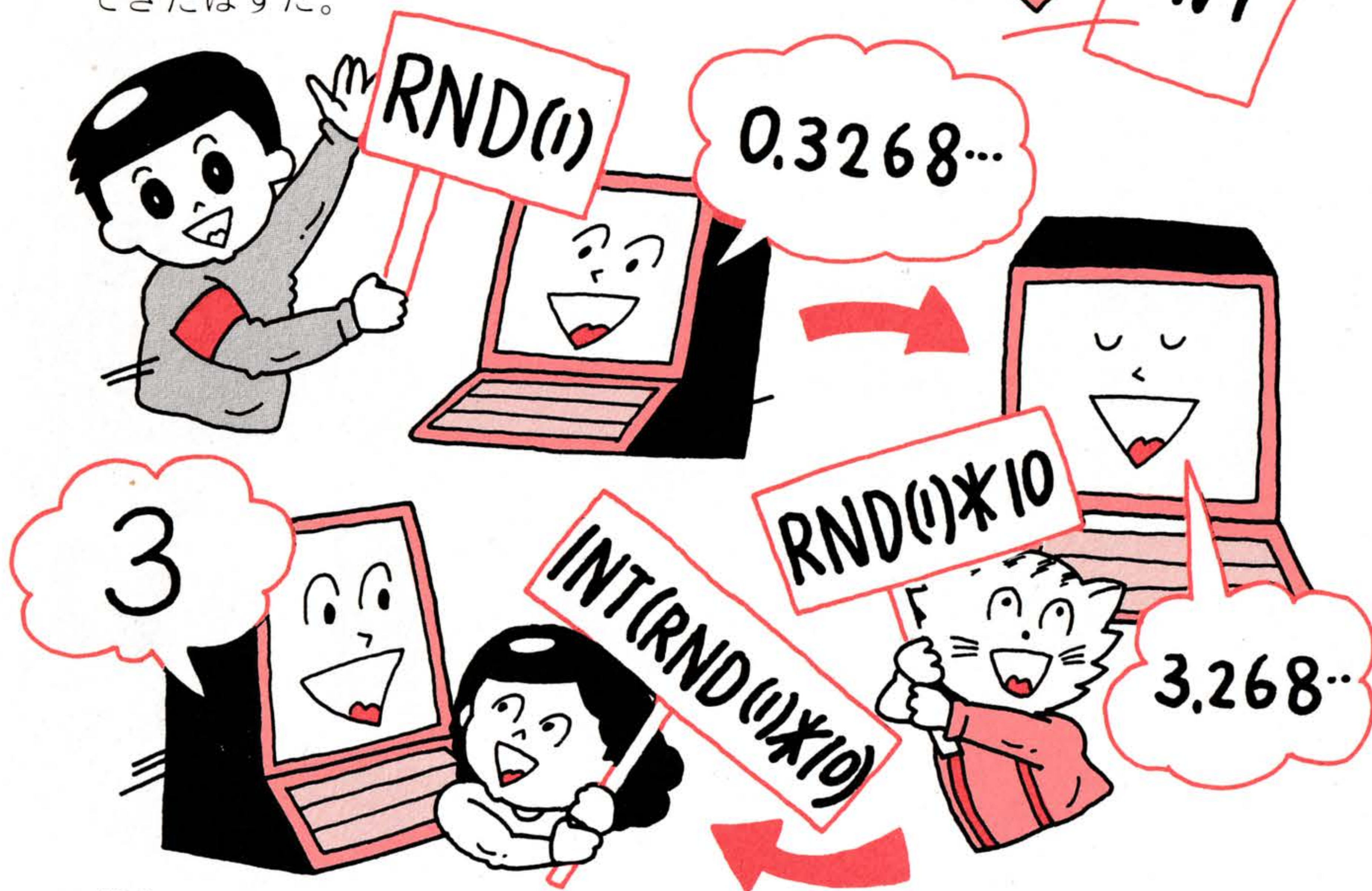
RETURN キーを押して実行してごらん。

さあ、これで、0より大きくて10より小さい整数、つまり0から9までの数がどんどん画面に出てきたはずだ。

2.57
4.63

整数になったけど、
この数は乱数だから
順番には出てこない
よ

5.2
↑
INT





1 から6までの数を作る

きみもよく知っているサイコロの目を出してみよう。1 から6 までの数だ。

まず、0 より大きくて6 より小さい数を作る。

$RND(1) * 6$

これで0 より大きくて6 より小さい数ができる。

これでは0 から5 までの数しかできない。1 より大きく7 より小さい数を作らなければならないから、これに1 をたしてやる。

$RND(1) * 6 + 1$

まだ小数点以下の数がついたままだから、 INT 命令でとると、

$INT(RND(1) * 6 + 1)$

この命令を与えてやると、コンピュータの中で1 から6 までの乱数が作られる。

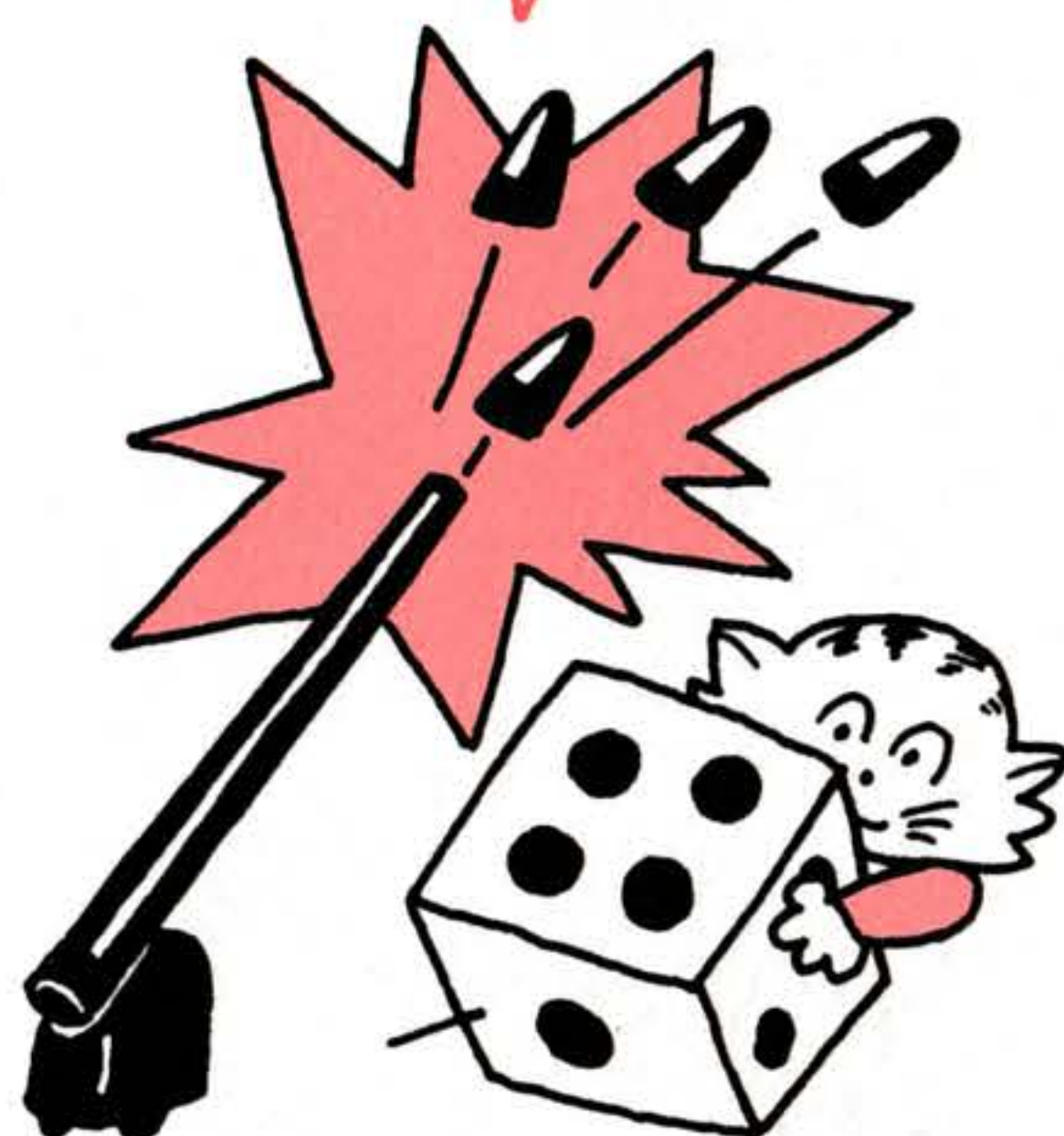
● 1 から6 までの数を作るプログラム

```
10 CLS
20 A=INT(RND(1)*6+1)
30 PRINT A
40 GOTO 20
```

● 1 から6 までの数をあてるゲーム

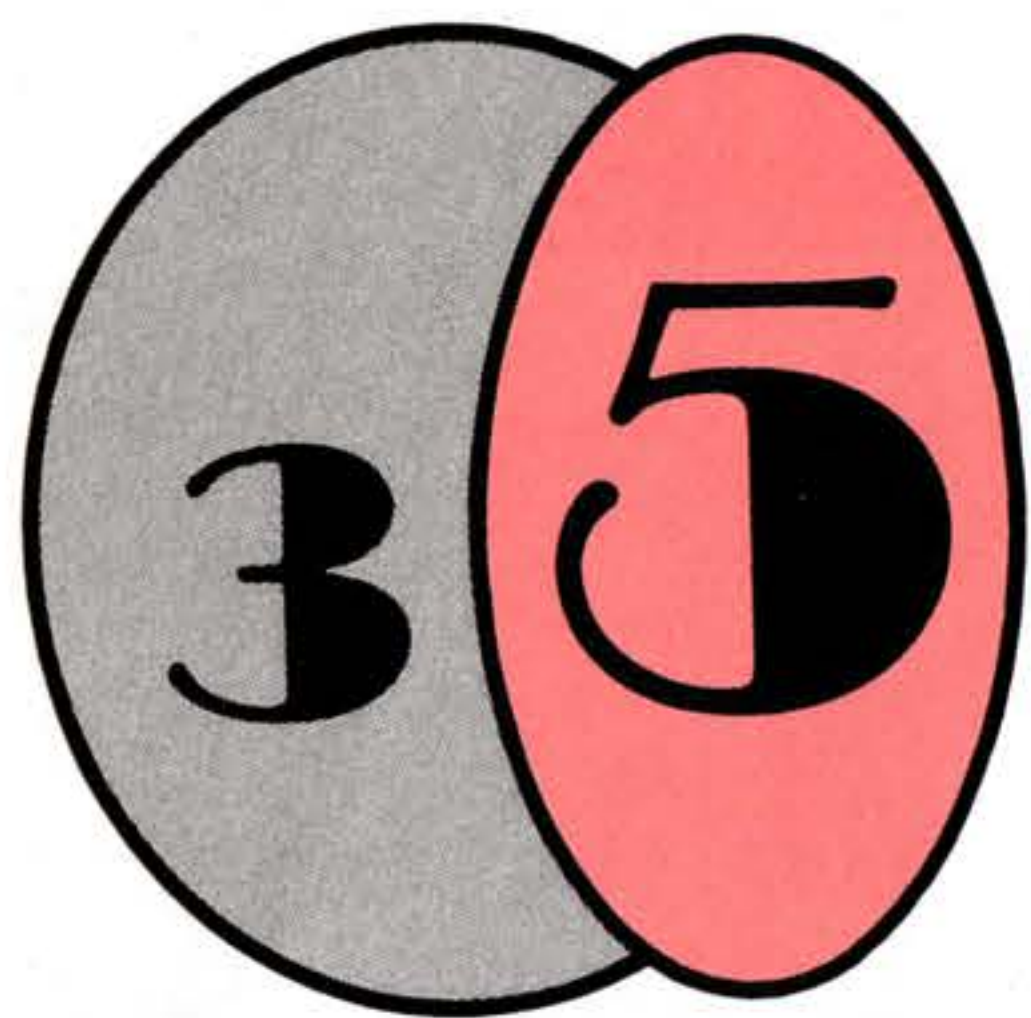
```
10 A=INT(RND(1)*6+1)
20 INPUT "1から6までのキーを押してサイコロの目を入力してください" : B
30 IF A=B THEN PRINT "アタリ!!"
   ELSE PRINT "ハズレ!!"
40 GOTO 10
```

RND と INT は、数に
関係するいろいろな
ところで使われているよ。何回ゲームが
できるとか、キャノ
ン砲を何発出すよう
にするとか、ネ



キーボードから、きみの好きな
数を入力してみよう。コンピ
ュータの持っている数と、きみの
入力した数があれば、“アタリ”
と表示されるよ



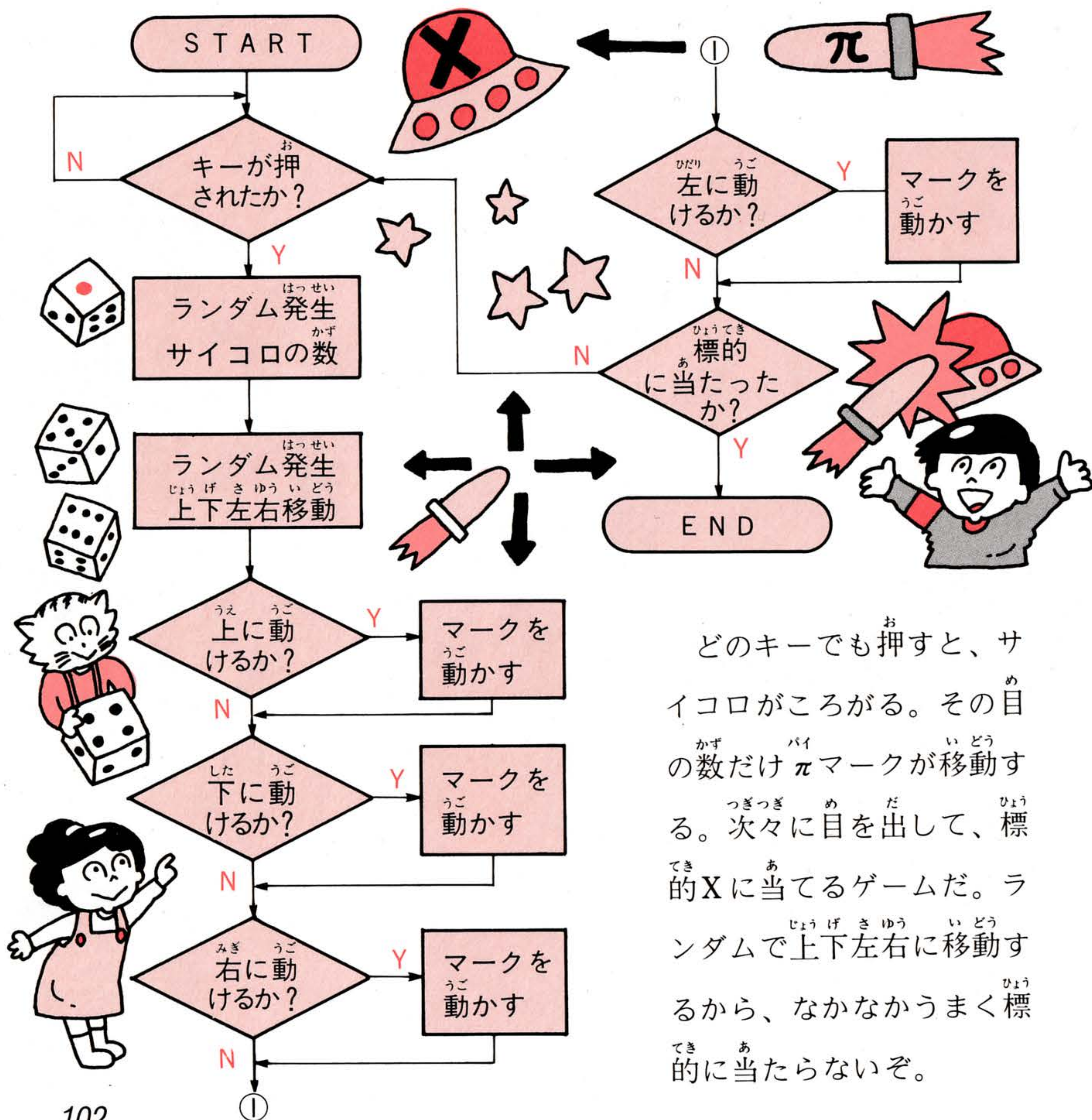


サイコロ応用ゲーム



パイ い どう ひょうてき あ
πマークを移動して標的に当てる

まず、フローチャートからだ。



どのキーでも押すと、サイコロがころがる。その目の数だけπマークが移動する。次々に目を出して、標的に当てるゲームだ。ランダムで上下左右に移動するから、なかなかうまく標的に当たらないぞ。

●ゲームプログラム サイコロコロコロ

```

10 CLS
20 SCREEN 0,0
30 WIDTH 40:KEY OFF
40 FOR Y=1 TO 3
50 LOCATE 0,Y
60 FOR X=1 TO 40
70 PRINT" ";
80 NEXT X,Y
90 PRINT"_____";
100 PRINT"_____]"
110 FOR N=4 TO 17
120 PRINT"|"
130 NEXT
140 PRINT"_____";
150 PRINT"_____]"
160 X1=6:Y1=6
170 LOCATE 20,11:PRINT"x"
180 LOCATE X1,Y1:PRINT"π"
190 S=0:LOCATE 12,2
200 PRINT"SAIKORO KOROKORO"
210 K$=INKEY$:IF K$=""THEN 210
220 A=INT(RND(-TIME)*6)+1
230 BEEP:S=S+1
240 IF K$=" " THEN PRINT"GOOD BY!":END
250 X=30:Y=19
260 ON A GOSUB 480,540,600,660,720,780
270 IF S=15 THEN 290
280 GOTO 220
290 C=INT(RND(-TIME)*6)+1
300 LOCATE X1,Y1:PRINT" "
310 IF (C=2)OR(C=3) THEN X1=X1+A
320 IF C=5 THEN X1=X1-A
330 IF C=6 THEN Y1=Y1-A
340 IF (C=1)OR(C=4) THEN Y1=Y1+A
350 IF X1<=2 THEN X1=2
360 IF X1>=37 THEN X1=37
370 IF Y1<=5 THEN Y1=5
380 IF Y1>=18 THEN Y1=18
390 LOCATE X1,Y1:PRINT"π"
400 IF (X1=20)AND(Y1=11) THEN 420
410 S=0:GOTO 210
    
```

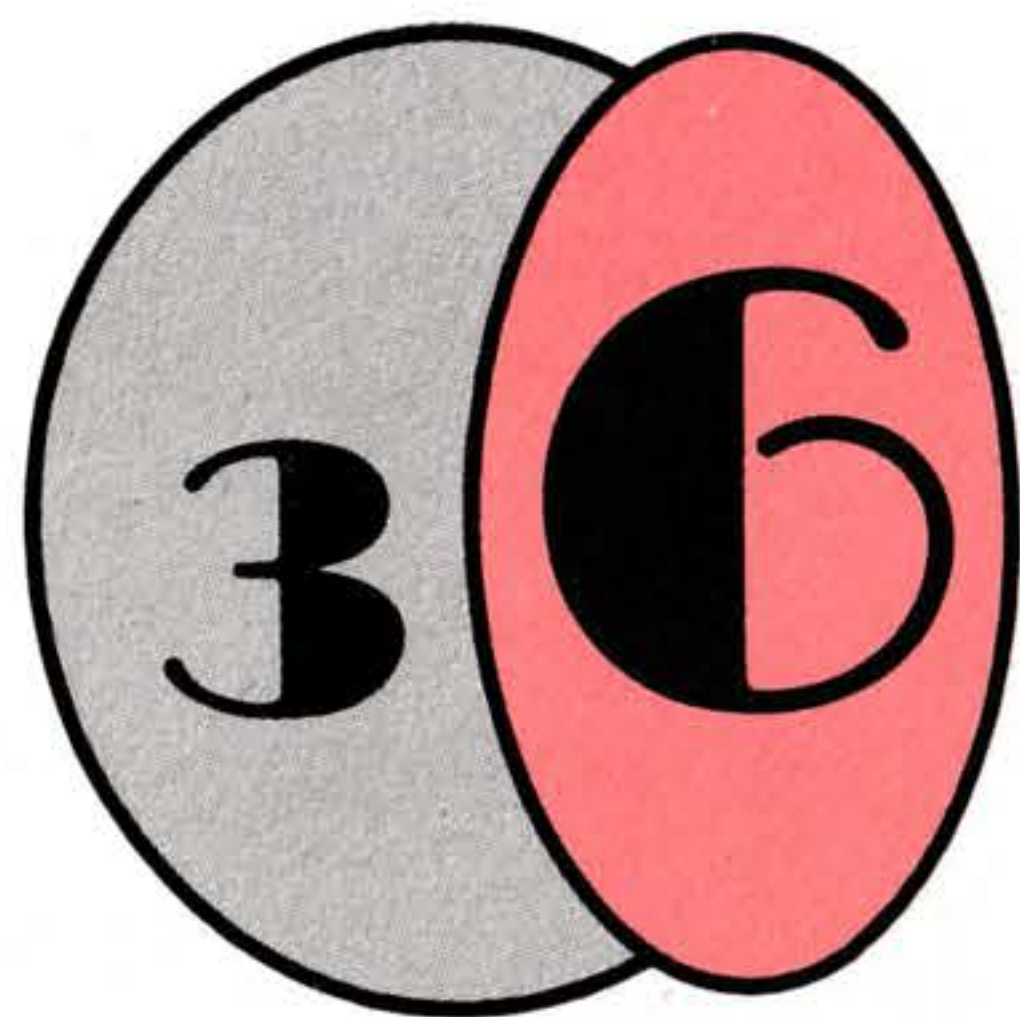


```

420 LOCATE 19,10:PRINT"END"
430 FOR M=1 TO 20
440 BEEP:LOCATE 19,10:PRINT"      "
450 FOR T=1 TO 100:NEXT
460 LOCATE 19,10:PRINT"END"
470 NEXT:END
480 LOCATE X,Y: PRINT"  _____  "
490 LOCATE X,Y+1:PRINT"|          |"
500 LOCATE X,Y+2:PRINT"|  ●  |"
510 LOCATE X,Y+3:PRINT"|          |"
520 LOCATE X,Y+4:PRINT"  _____  ";
530 RETURN
540 LOCATE X,Y: PRINT"  _____  "
550 LOCATE X,Y+1:PRINT"| 0  |"
560 LOCATE X,Y+2:PRINT"|          |"
570 LOCATE X,Y+3:PRINT"|  0 |"
580 LOCATE X,Y+4:PRINT"  _____  ";
590 RETURN
600 LOCATE X,Y: PRINT"  _____  "
610 LOCATE X,Y+1:PRINT"| 0  |"
620 LOCATE X,Y+2:PRINT"|  0 |"
630 LOCATE X,Y+3:PRINT"|  0 |"
640 LOCATE X,Y+4:PRINT"  _____  ";
650 RETURN
660 LOCATE X,Y: PRINT"  _____  "
670 LOCATE X,Y+1:PRINT"| 0 0 |"
680 LOCATE X,Y+2:PRINT"|          |"
690 LOCATE X,Y+3:PRINT"| 0 0 |"
700 LOCATE X,Y+4:PRINT"  _____  ";
710 RETURN
720 LOCATE X,Y: PRINT"  _____  "
730 LOCATE X,Y+1:PRINT"| 0 0 |"
740 LOCATE X,Y+2:PRINT"|  0 |"
750 LOCATE X,Y+3:PRINT"| 0 0 |"
760 LOCATE X,Y+4:PRINT"  _____  ";
770 RETURN
780 LOCATE X,Y: PRINT"  _____  "
790 LOCATE X,Y+1:PRINT"| 000 |"
800 LOCATE X,Y+2:PRINT"|          |"
810 LOCATE X,Y+3:PRINT"| 000 |"
820 LOCATE X,Y+4:PRINT"  _____  ";
830 RETURN

```





さんすうドリル プログラム



もんちゅう 10問中いくつあたったか？

かんたんな計算問題のドリルを作ってみよう。
問題はひと桁のかけ算だ。仕事の流れを考えてから、フローチャートを書いてみよう。

プログラミングの流れ

1. コンピュータが問題を作って、画面に表示する。
2. その問題の答えを、きみがキーボードから入力する。
3. 答えがあっているかどうかを、コンピュータが判断する。答えがあていば○、まちがていば×を表示する。
4. 問題が10問になると、終わる。

フローチャートは、これを図に直せばいいだけだ。それほどむずかしいプログラムではないから、先に進む前に、トライしてみてもいいよ。

ポイントとなる命令は、問題を作るために数を発生させるRND、INTと、アタリ、ハズレを判定するIF～THEN命令みたいだね。

フローチャートに書くまえに、どんな命令を使ったら、このプログラムが作れるのかを自分で考えてみよう



乱数を使えば、どんな問題が出るのかきみにもわからないぞ



●さんすうドリルのフローチャート

3+2 6+1

10問の問題を発生させるループ。FOR～NEXTだ。10問出し終わったら、ENDだ。

5+2=6



START

問題をランダムで発生させる

ひと桁の計算問題だから、1から9までの数だ。作ってみよう。

答えをキー入力

INPUT命令だね。

入力した答えはあっているか?

IF～THENが出てきたぞ。

YES

画面に○を表示

NO

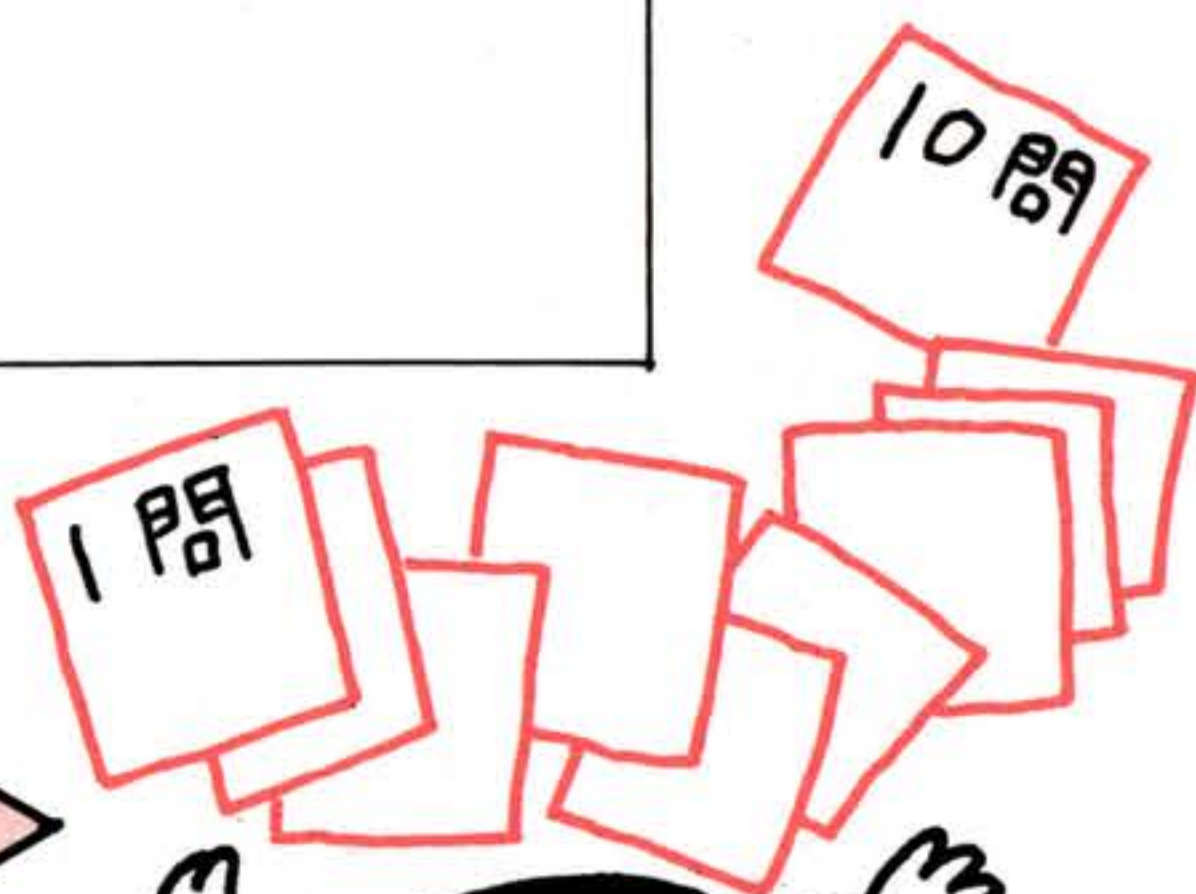
画面に×を表示

NO

10問になったか?

YES

END



このフローチャートで、たし算、ひき算、かけ算、わり算の4種類の計算問題を作れる。桁数をふやせば、むずかしい問題も出せるぞ

●さんすうドリルのプログラム

```

10 CLS
20 SCREEN 0,0
30 WIDTH 40
40 KEY OFF
50 LOCATE 15,0:PRINT"カケサツン"
60 FOR N=1 TO 10
70 X=INT(RND(-TIME)*9+1)
80 Y=INT(RND(-TIME)*9+1)
90 LOCATE 8,1+N*2
100 PRINT X;"*";Y;"="
110 LOCATE 17,1+N*2:INPUT A
120 IF X*Y=A THEN LOCATE 23,1+N*2:S=S+1
130 IF X*Y=A THEN PRINT"O":NEXT:GOTO160
140 LOCATE 23,1+N*2
150 PRINT" X ":NEXT
160 LOCATE 27,18
170 PRINT"10モン チュウ "
180 LOCATE 26,19
190 PRINTS;"モン アタリテ"ス"
200 LOCATE 27,21
210 PRINT"キー ラ オス"
220 Z$=INKEY$:IF Z$="" THEN GOTO 220
230 RUN
    
```



もんだい だ
問題を出して、O×
はんてい
を判定するループ



●プログラムの説明

かんたんにプログラムの説明をしておこう。

10行～50行

初期設定^{しよ き せつてい}といって、画面表示^{が めんひょうじ}の準備^{じゅんび}をする部分^{ぶ ぶん}だ。20行のSCREEN 0, 0で、文字表示用の画面^{も じひょうじよう が めん}を指定^{し してい}して、WIDTH 40で、横^{よこ}に40字表示^{じひょうじ}できるようにした。

40行の「KEY OFF」は、初めて出てくる命令^{めいれい}だね。電源^{でんげん}を入れたときの画面^{が めん}には、下^{した}のほうに

キー オフ
KEY OFFは、

ファンクションキー
の内容^{ないよう}を画面^{が めん}から消
す命令^{めいれい}だ



ファンクションキーの内容が表示されているね。
これはべつに^{ひょうじ}表示されているままだもかまわない
けれど、ないほう^{がめん}が画面が見やすくなるというとき
には、^{キー}KEY ^{オフ}OFF命令^{めいれい}を実行^{じっこう}すると消^きえてしま
う。

60行^{ぎょう}～150行^{ぎょう}

この部分^{ぶぶん}のFOR～NEXT^{フォーネクスト}ループで、10回^{かいもんだい}問題を
出^だしているのがわかるかな？

：（コロン）でつな
ぐマルチステートメ
ントがいくつか出^で
てくるから注意^{ちゅうい}しよう



```
60 FOR N=1 TO 10
70 X=INT(RND(-TIME)*9+1)
80 Y=INT(RND(-TIME)*9+1)
90 LOCATE 8,1+N*2
100 PRINT X;"*";Y;"="
110 LOCATE 17,1+N*2:INPUT A
120 IF X*Y=A THEN LOCATE 23,1+N*2:S=S+1
130 IF X*Y=A THEN PRINT"0":NEXT:GOTO 160
140 LOCATE 23,1+N*2
150 PRINT "X":NEXT
```

かけ算^{さん}をする2つの数^{かず}をX、Yの変数^{へんすう}として、
1から9までの整数^{せいすう}をランダムに発生^{はっせい}させている
のが70行^{ぎょう}と80行^{ぎょう}だ。

ここで（-TIME^{タイム}）という見^みなれないものが出^で
てきたね。1から9までの整数^{せいすう}を発生^{はっせい}させるには、

`INT(RND(1)*9+1)`

でも、もちろんいい。

ただ、この場合^{ばあい}は乱数^{らんすう}を発生^{はっせい}させるたびに同じ
ような値^{あたい}が出^でてくるんだ。だから同じ問題^{おなもんだい}が何回^{なんかい}
も出^でてきてしまうこともある。

ランダム
RND(1)だと、あ
れ、何回^{なんかい}か乱数^{らんすう}を出^だ
していると、同じパ
ターンに。RND(-
タイム)なら、まっ
たく不規則^{ふきそく}だ



ランダム (TIME) とすると、カッコ内の数を自動的に変えて、毎回発生させる乱数を変えられるんだ。だから、いろいろな問題が出題される。

120行と130行はIF文で答えが当たったときの処理だ。X * Yの答えがAと一致しないときには、140行から150行へ行って、まちがったときの処理、つまりXを画面に表示させるよ。

160行～190行

10問中いくつ当たったかを画面に表示する。

200行～230行

「キー ヲ オス」と画面表示が出たら、キーボードから何でも好きなキーを押せば、もう一度、10問の計算問題がスタートする。

$$5+1=6 \bigcirc$$

$$6+5=9 \times$$

120行の $S = S + 1$ は当たったときの回数をたしている。
○印を表示してから160行に飛ぶよ。
変数Sは190行にあるね



文字をつなげて表示させる； (セミコロン)

100行を見てみよう。Xは70行で発生させた乱数。Yは80行で発生させた乱数だ。；は、文字をつなげて表示するときに使う記号だ。

たとえばXが9、Yが5のときには、 $9 * 5$ と表示される。

110行では、=のうしろにカーソルを指定して、INPUTの？マークを表示させて、右の画面のような計算問題ができる。

；を、(コンマ)にかえると、次の文字との間に13文字分の空白をあけてしまう。

$$9 * 5 = ?$$

答えをキー入力して
RETURN キーだ





たし算、ひき算、わり算のドリル

●たし算のプログラムに変える

100行と120行、130行のアスタリスク（*）を
+（プラス）の記号に変えればOK。

問題をむずかしくしたければ、70行と80行の9
をもっと大きな数に変えればいい。

●ひき算のプログラムに変える

100行、120行、130行を-（マイナス）に変え
るのはたし算と同じだ。ただし、このままだと引
く数が引かれる数より大きくなって、-（マイナ
ス）の答えが出ることもあるよ。

マイナスの答えを出したくない場合は、プログ
ラムを1行追加する。80行のあとに、

85 IF X < Y THEN 70

この意味は、YがXより大きいときは、もう一
度、数を発生させるために70行にもどるというこ
とだ。追加したらRETURNキーを押すよ。

●わり算のプログラムに変える

わり算のプログラムに変えるときには、もう1
行プログラムを追加する。86行として、

86 IF (X/Y) <> INT (X/Y) THEN 70

これは、わり切れる問題を出すためのプログラ
ムで、<>の記号は「等しくない」という意味だ。

追加したら、LIST命令で確認しておこう。

*（カケル）を、+
（タス）、-（ヒク）、
/（ワル）に変える



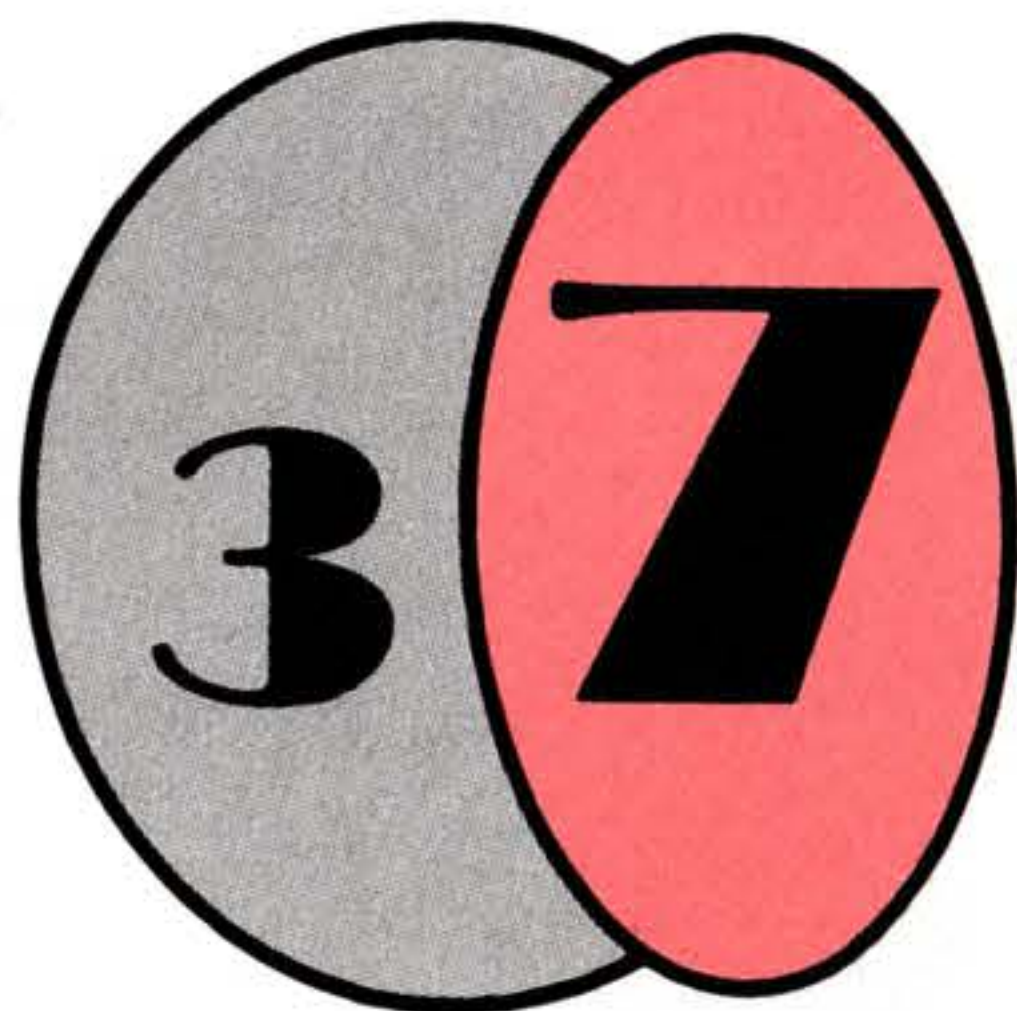
```
LOCATE 8, 1
.00 PRINT X;"*";
110 LOCATE 17, 1+
120 IF X*Y=A THE
130 IF X*Y=A THE
'40 LOCATE 23, 1
```

ひき算、わり算は、
ちょっと工夫が必要



追加したらかならず
RETURNキー。
LISTして確認す
ることも忘れずに。





プログラムの整理



行を加えたりけずったりしたとき

RENUM命令 ● プログラムの行番号を整理する

RENUM命令は、行番号をそろえる命令だ。

わり算プログラムに変えたときを考えてみよう。

AUTO命令で、10番きざみにきれいに並んでいたプログラムが、途中から変わってしまったね。

もう一度10番きざみの行番号にして整理しよう。

RENUM

とキー入力して、**RETURN**キーを押す。

画面上のプログラムはなにも変化しないけれど、

RENUM表示のすぐ下にOKと表示されているはずだ。

RENUM命令は、「プログラムの行番号を付け変えろ」という命令だけで、整理されたプログラムを表示させるのは、LIST命令だ。

プログラムを整理するときは、LIST命令で確認しよう。

プログラムを整理するときは、LIST命令で確認しよう。

DELETE命令 ● プログラムを行番号ごとにとってしまう

DELETE命令は、プログラムを行番号ごとにとってしまう命令だ。

2～3行消すだけだったら、消したい行番号をキー入力して、**RETURN**キーを押せば消えてしまう。

DELETE命令は、たくさんのプログラムをまとめて消すときに便利なんだ。

DELETE命令は、たくさんのプログラムをまとめて消すときに便利なんだ。

DELETE命令は、たくさんのプログラムをまとめて消すときに便利なんだ。

10番きざみにつけた行番号のあいだには、9個の行番号が使える。追加したあと、RENUMで10番きざみに戻しながら、いくらかでもプログラムを追加できるぞ



一度消すともとにもどらないから慎重に

つか かた
使い方いろいろある。DELETEとキー
にゅうりょく
入力したら、RETURNキーを忘れずに。終わっ
たら、同じくRENUMで整理して、LIST命令で画
めん だ かくにん
面に出して確認しよう。



DELETE 行番号……この行番号だけを消す
DELETE 行番号①—行番号②……①から②までの行を消す
DELETE 行番号—……この行番号からあとの行を消す
DELETE —行番号……最初の行からこの行番号までを消す

●プログラムの編集をするRENUMとDELETE

①
60 FOR N=1 TO10
70 X=INT(RND(-TIME)*9
80 Y=INT(RND(-TIME)*9
85 IF X<Y THEN 70
86 IF(X/Y)<>INT(X/Y)
90 LOCATE 8,1+N*2
100 PRINT X;"*";Y;"="
110 LOCATE 17,1+N*2:I

②
60 FOR N=1 TO10
70 X=INT(RND(-TIME)*9
80 Y=INT(RND(-TIME)*9
90 IF X<Y THEN 70
100 IF(X/Y)<>INT(X/Y)
110 LOCATE 8,1+N*2
120 PRINT X;"*";Y;"="
130 LOCATE 17,1+N*2:I



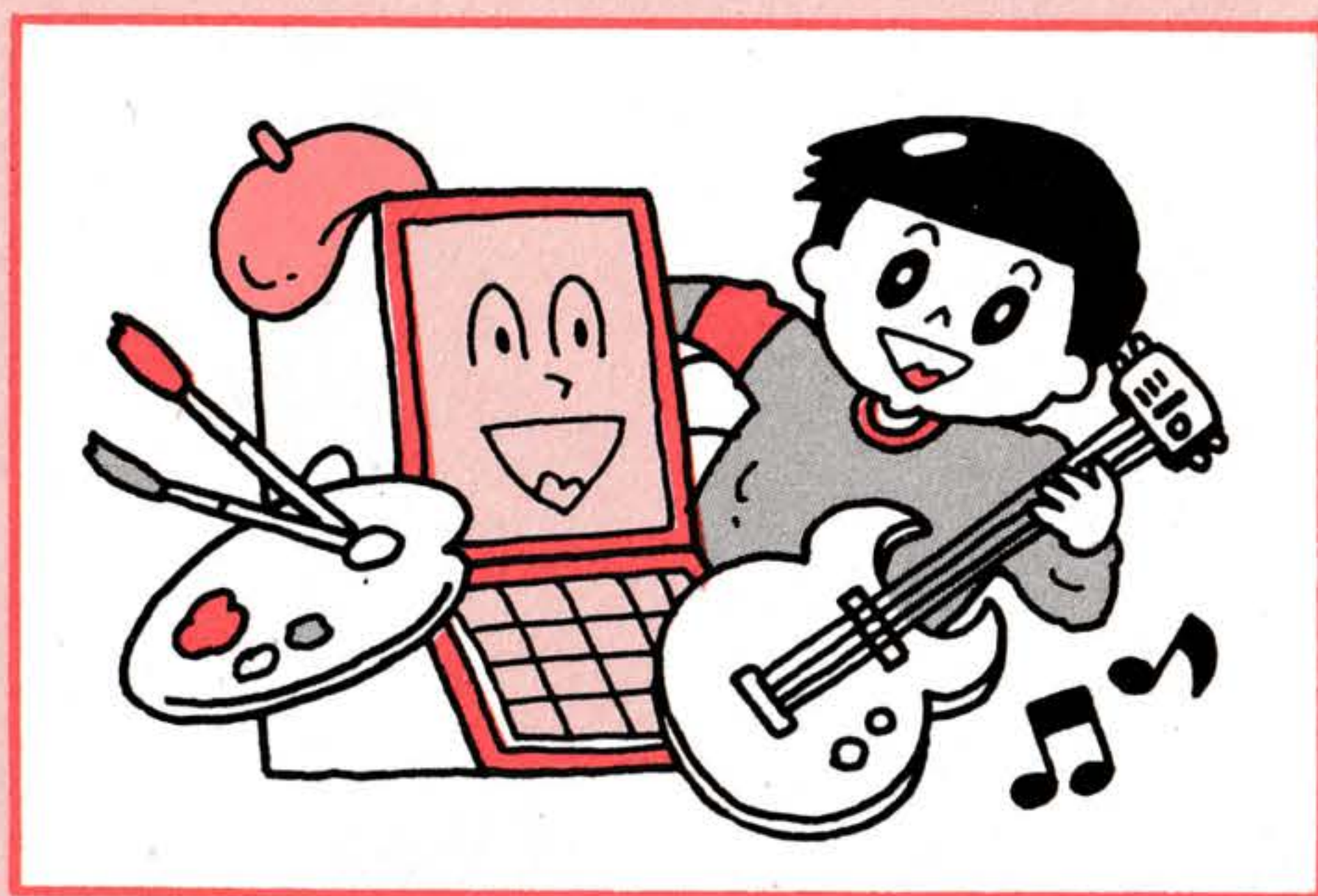
③
10 LOCATE 10,10
20 PRINT"ABCDEF"
30 LOCATE 4,10
40 PRINT"*****"
50 PRINT"GHI"
60 END

④
10 LOCATE 10,10
20 PRINT"ABCDEF"
60 END



パート4

図や絵を描いたり 音を出したり



4

1

グラフィックの 画面作りでスタート



スクリーン え か
SCREEN2と3は絵を描くモード

スクリーン めいれい が めん き
SCREEN命令 ● どんな画面にするかを定める

パソコンに図や絵を描かせて、色を塗ったりしてみよう。図や絵を描く場所は、もちろんディスプレイだ。でも、電源をONしたとき、ディスプレイの画面は、字を書くための画面になっているから、まず、絵が描ける画面作りからスタートだ。画面作りの命令は、SCREENだ。

SCREEN 画面モード

という形で使う。「画面モード」というところには0～3までの数字を入れる。0と1はテキストモードだから、絵を描くなら、2か3だ。SCREEN2で画面を作ってみよう。

```
10 SCREEN 2
20 PSET(130,100),15
30 GOTO 30
```

このかんたんなプログラムを入力して実行してみよう。画面のまん中あたりに、小さな小さな白い点が表示されたはずだ。20行は、(130, 100)の位置に白い色(15)で点を描けということだよ。

パソコン
グラフィックスと
いうんだ



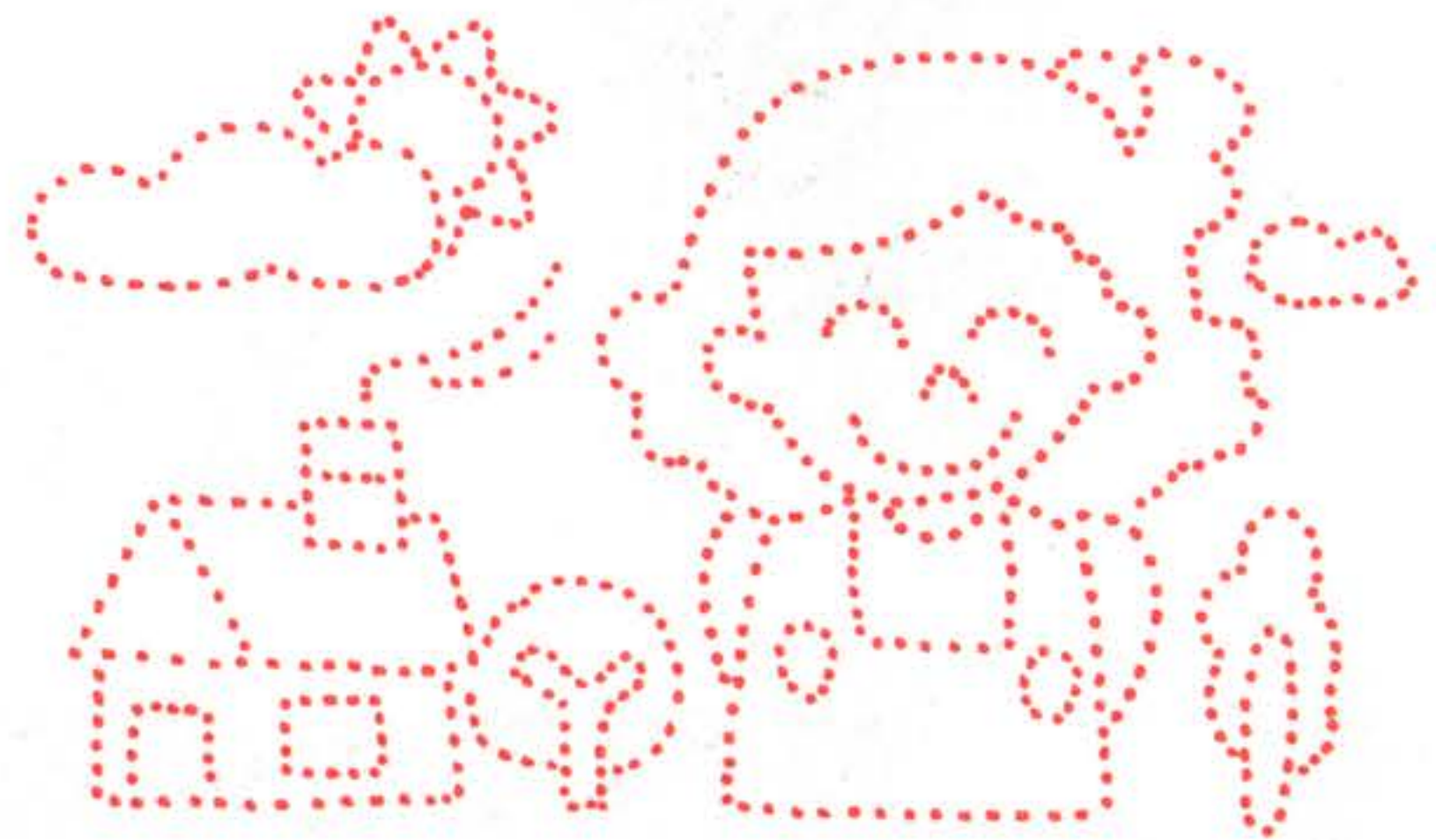
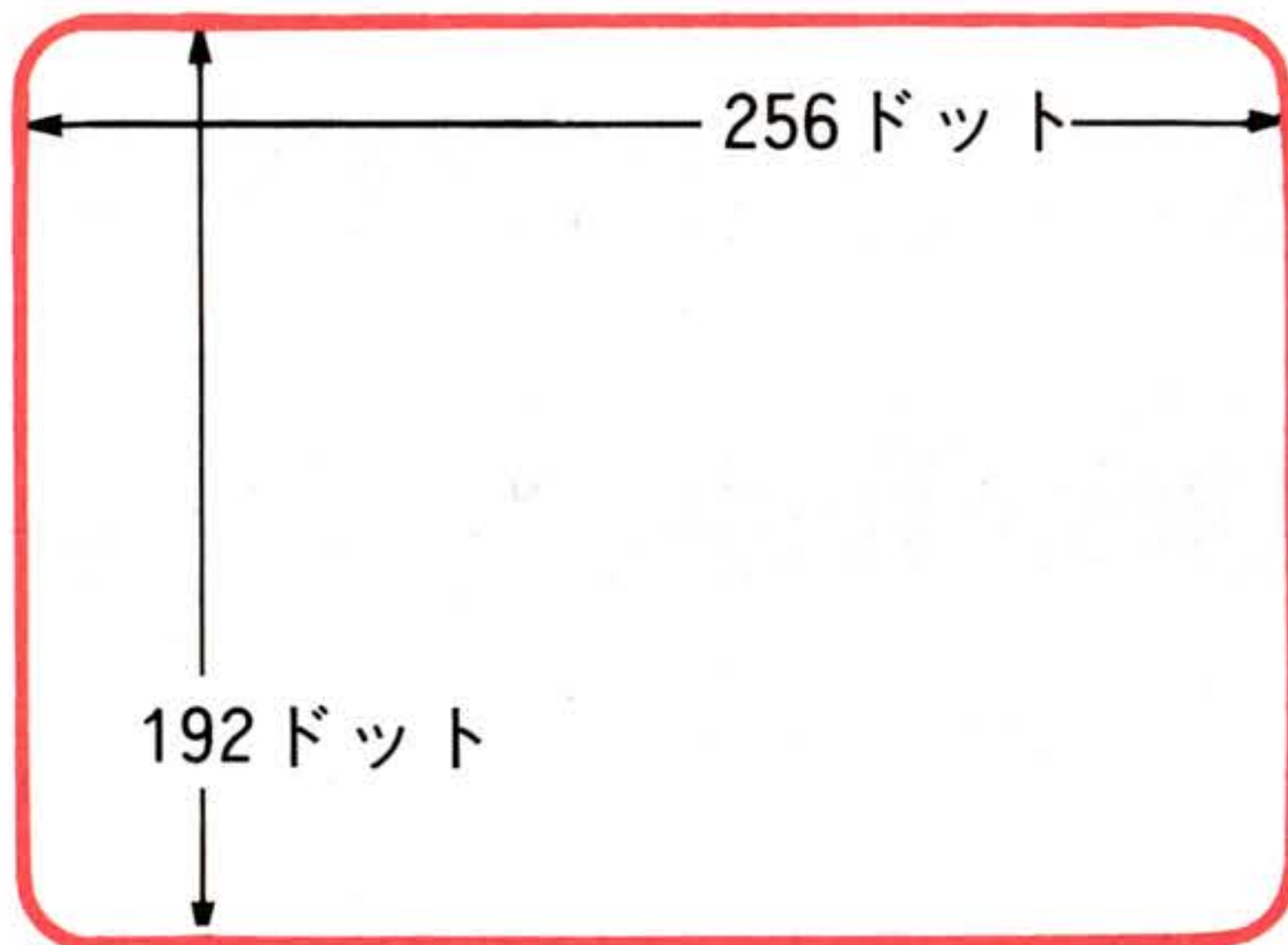
テキストモードは
字を書く画面
グラフィックモードは
絵を描く画面よ



●
白い点

●グラフィック画面

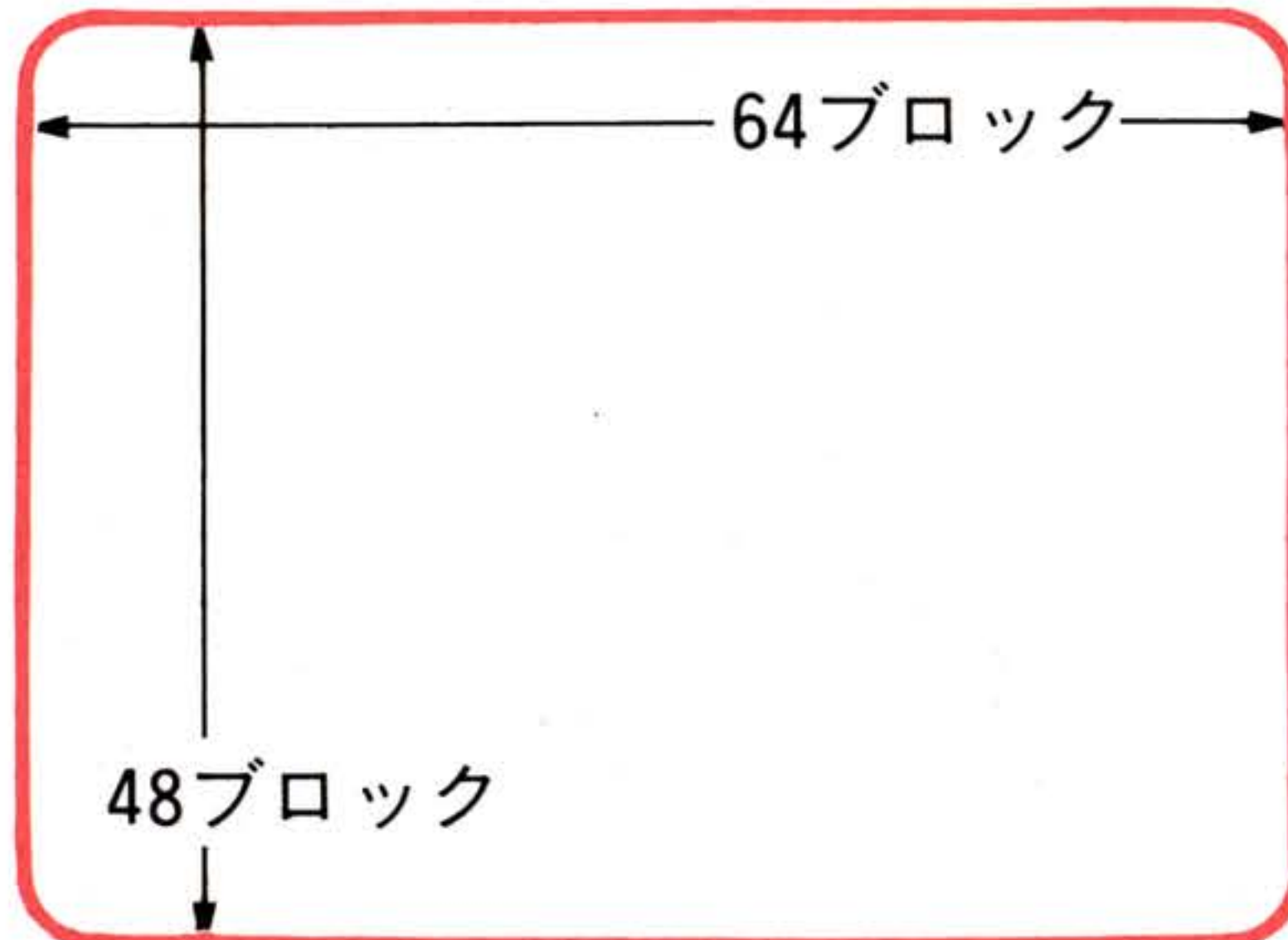
●SCREEN 2



こうかいぞうど
高解像度グラフィックモード。

こま かい え か
細かい絵を描くことができるが、
いろ つか かた せいやく
色の使い方に制約がある。

●SCREEN 3



マルチカラーモード。1 ブロッ
クの大きさは、SCREEN 2 にく
らべて縦横 4 倍だから、大まかな
絵になる。

スクリーンめいれい
SCREEN命令

スクリーンめいれい してい ぜんぶ ようそ してい つぎ
SCREEN命令で指定できる全部の要素を指定すると、次のような
ぶんけい
文型になる。

SCREEN モード, スプライトサイズ, キークリックス
イッチ, カセットボーレート, オプション

スプライトサイズ: 0 ~ 3 でスプライトの大きさ / キークリック
スイッチ: 0 は音を出さない、1 は音を出す / カセットボーレート
: カセットに書き込む速度。1 は1200ボー、2 は2400ボー / オプシ
ョン: 0 はMSX標準プリンタ、1 は標準以外のプリンタ

4

2

図や絵に色をつける



使用できる絵の具は15色

COLOR命令●前景色、背景色、周辺色を決める

グラフィック画面を作って、図や絵を描いても、白と黒のモノクロじゃおもしろくない。絵の具の色は15色あるのだから、自由自在に色をつけよう。

色をつける命令はCOLOR文だ。

COLOR 前景色、背景色、周辺色

という形で使う。前景色は表示した絵の点や線につける色。背景色は画面の地の色。周辺色は画面の中で文字や形を表示できない外側のふちどりの色だ。

前景色、背景色、周辺色とも、カラーコードにある数字で指定すればいい。

```
10 CLS
20 COLOR 13,15,1
30 SCREEN 2
40 LINE(20,20)-(150,100),,BF
50 GOTO 50
```

上のプログラムを入力して実行してみよう。紫で塗られた四角形が表示され、地の色は白で、ふちどりは黒になっただろう。

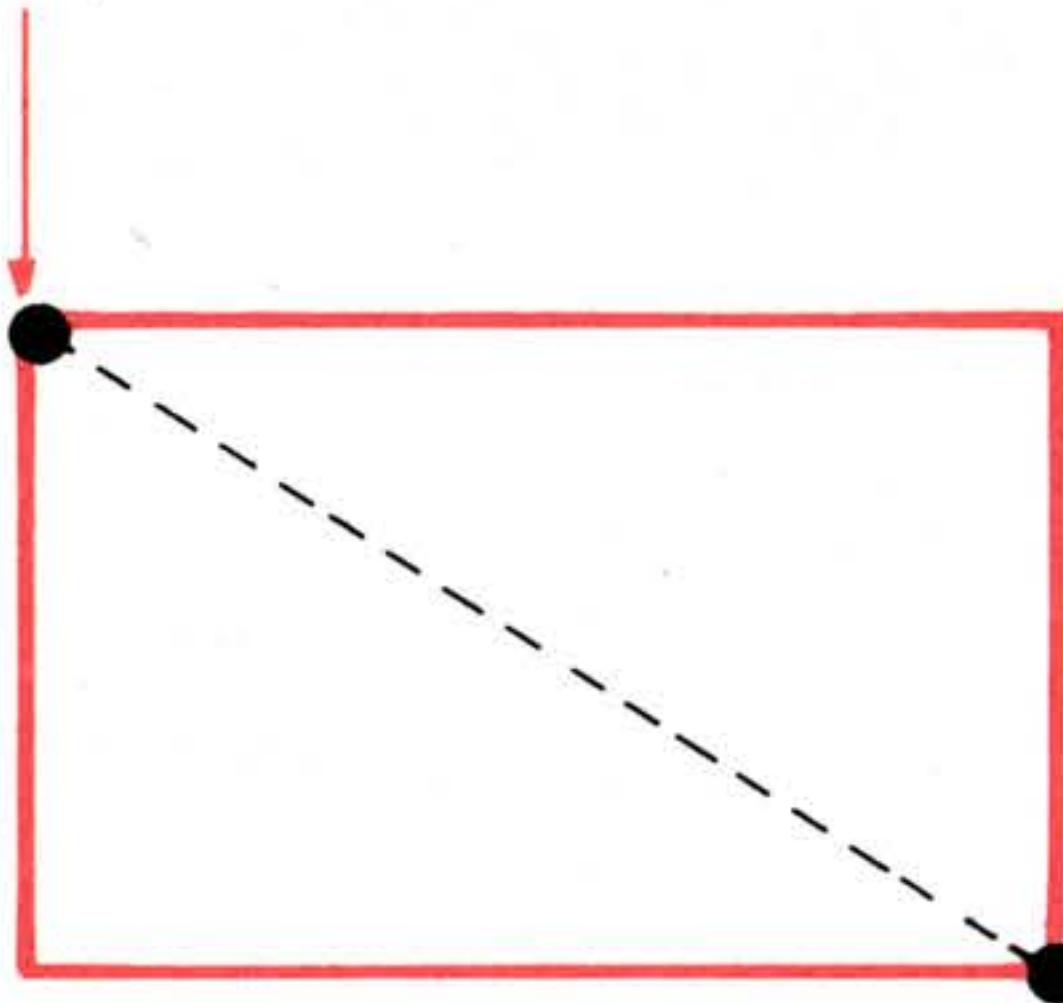
コード	色
0	周辺色と同じ色
1	黒
2	緑
3	明るい緑
4	暗い青
5	明るい青
6	暗い赤
7	水色
8	赤
9	明るい赤
10	暗い黄
11	明るい黄
12	暗い緑
13	紫
14	灰色
15	白

● 絵ときCOLOR命令

図形の表示は——

40 LINE (20, 20) — (150, 100) , , BF

画面の (20, 20) の位置



COLOR文と同じ表示色を使うときは省略してもいい

LINEは線を引いたり箱を描いたりする命令だ

画面の (150, 100) の位置



描いた図形に色をつけるのは——

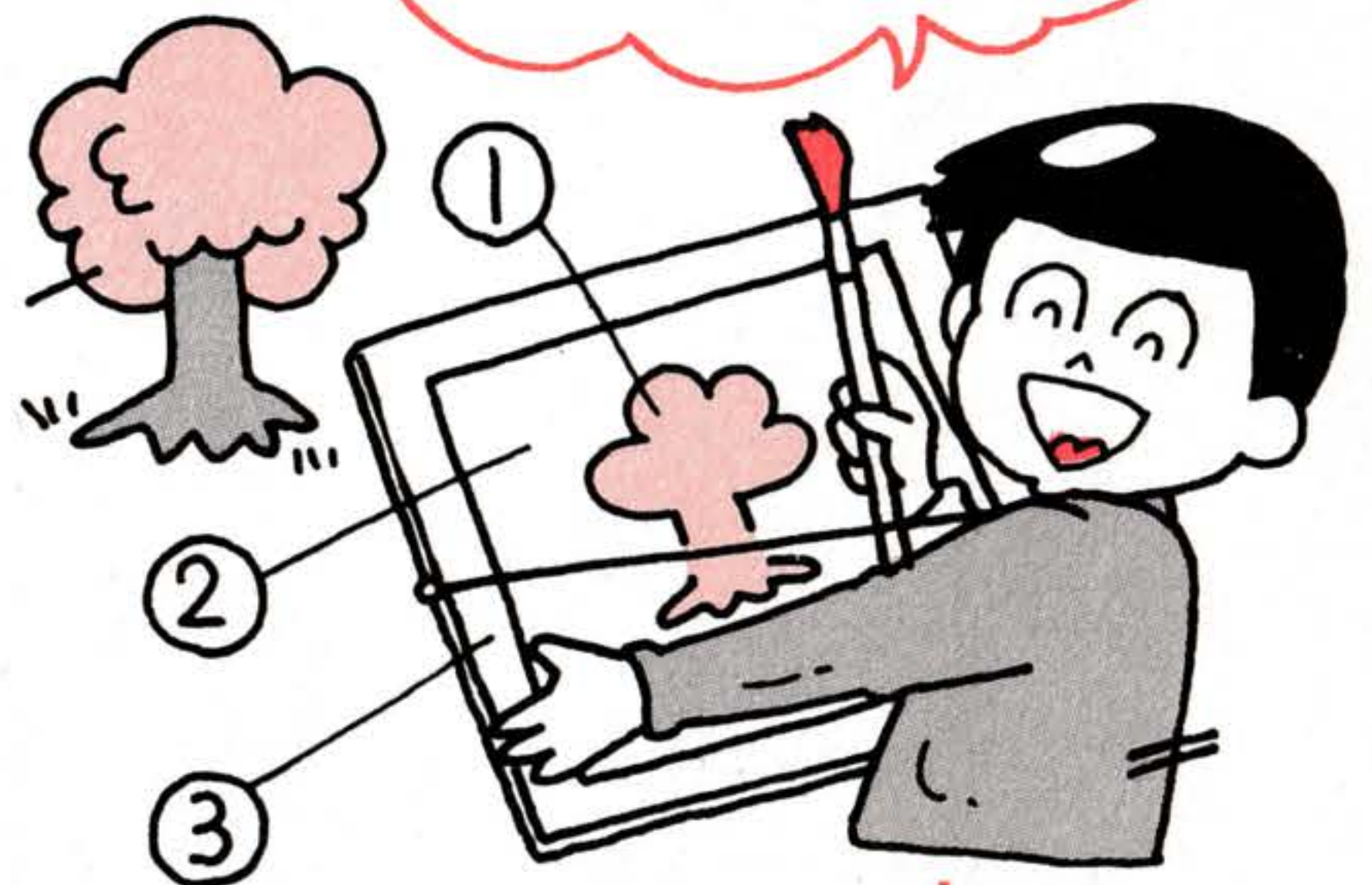
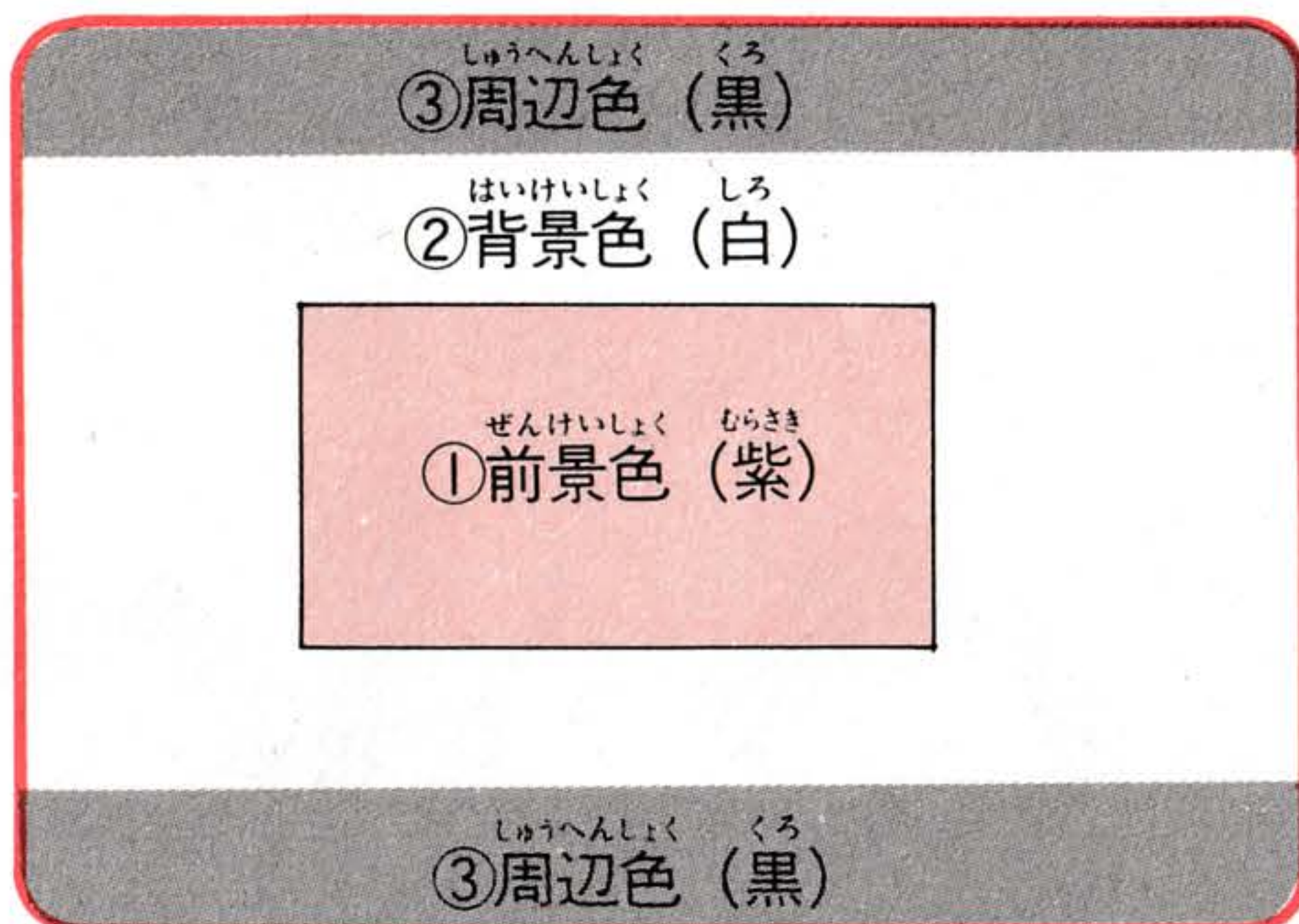
20 COLOR 13, 15, 1

外わくの色。1は黒だ
地の色。15は白だ
図形の色。13は紫だ

COLOR命令で指定の順は

ゼットアイに

①前景色 ②背景色 ③周辺色の順だよ



はじめにスイッチを入れたときは、前景色は白 (15)、背景色は暗い青 (4)、周辺色は水色 (7) になっているよ

●COLOR文ショートプログラム

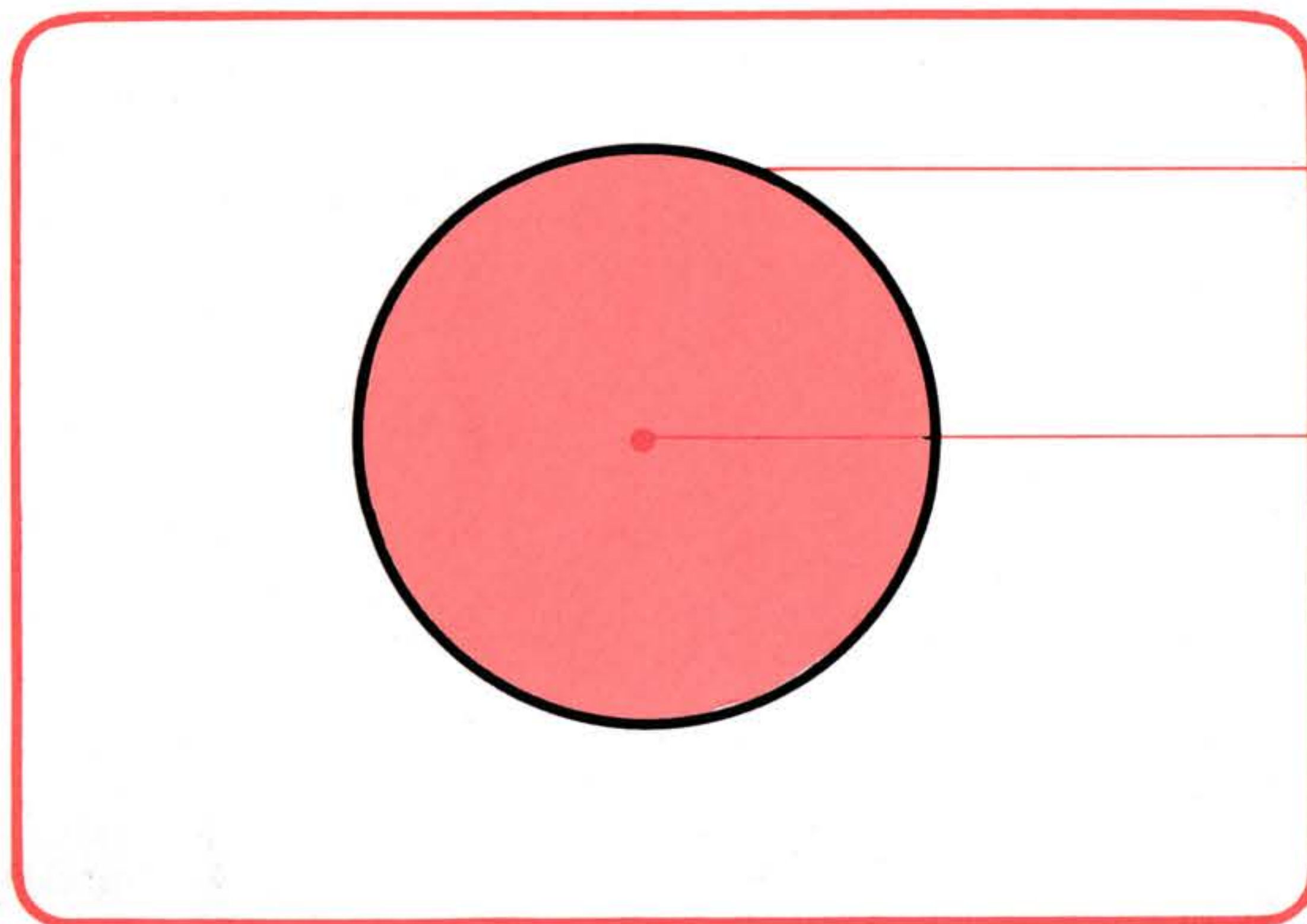
●カラーコードの14色をぜんぶ表示させてみる

```
10 COLOR , 1
20 SCREEN 2
30 FOR I=2 TO 15
40 CLS
50 COLOR I
60 CIRCLE(120,90),60
70 PAINT(120,90)
80 FOR K=1 TO 500
90 NEXT K,I
100 COLOR 15,4
110 END
```

カラーコードに入っている2
～15までの色をとり出して、円
の中に順番に表示させる。

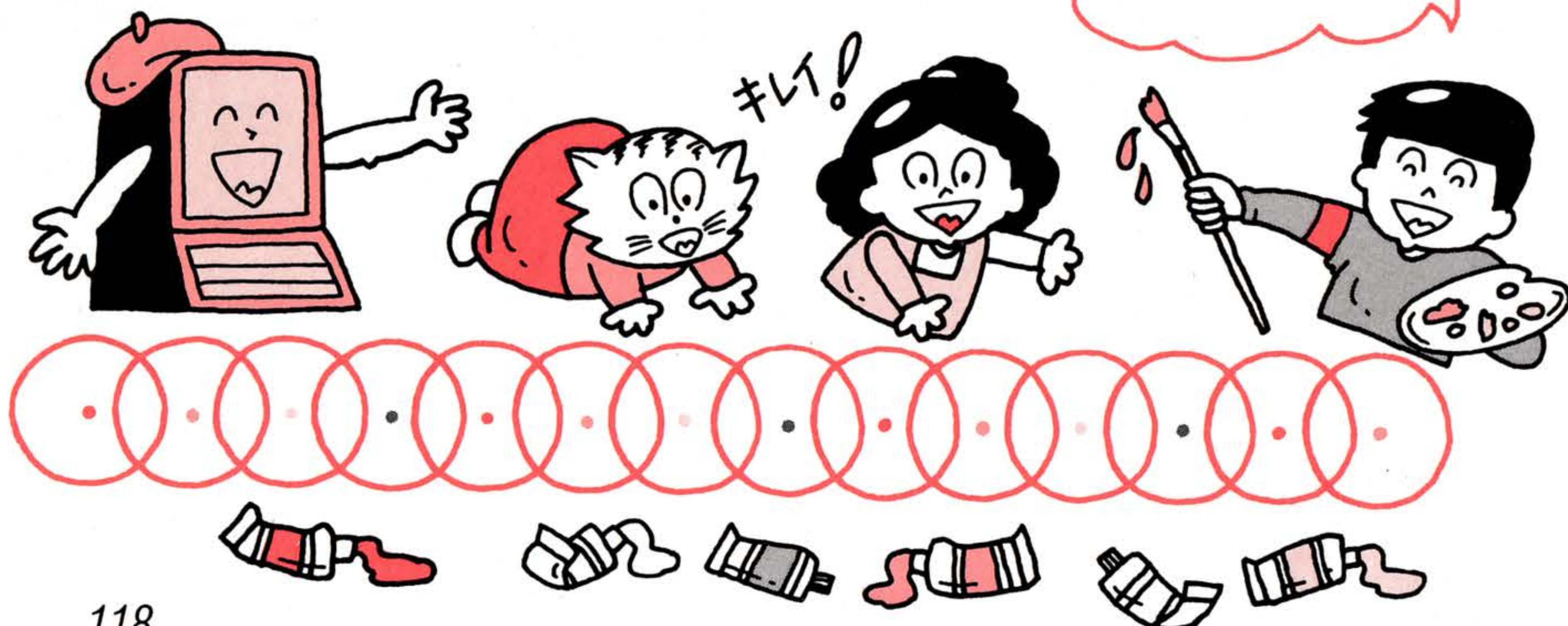
60行は円を描く命令。70行の
PAINT文は、そこを塗れという
命令だ。

30～90行のFOR～NEXTで、
14回色を変えている。



円を描くのはC I R
C L E 文。半径は60
ドットだ
円の中心座標は(120,
90)

円の中の色が
どんどん変わるよ



4

3

直線で描く グラフィックス



まず1本の線を引いてみよう

LINE命令 ●線を引き、箱を描き、箱を塗る

グラフィック画面作りもやったし、色を塗る方法もすました。キャンバスと絵の具の筆が用意できたのだから、こんどは筆で、線画を描く番だ。
まず、線を1本引いてみよう。

```
10 SCREEN 2
20 LINE(50,50)-(150,150),9
30 GOTO 30
```

画面の横50、縦50の位置にある点と、横150、縦150の位置にある点を結んで1本の線を引け。線の色は、カラーコード9の赤だよ、という命令だ。

線をどこに表示させるかは始点の位置と終点の位置を決めてやればいい

横の位置はX座標
縦の位置はY座標
というんだよ

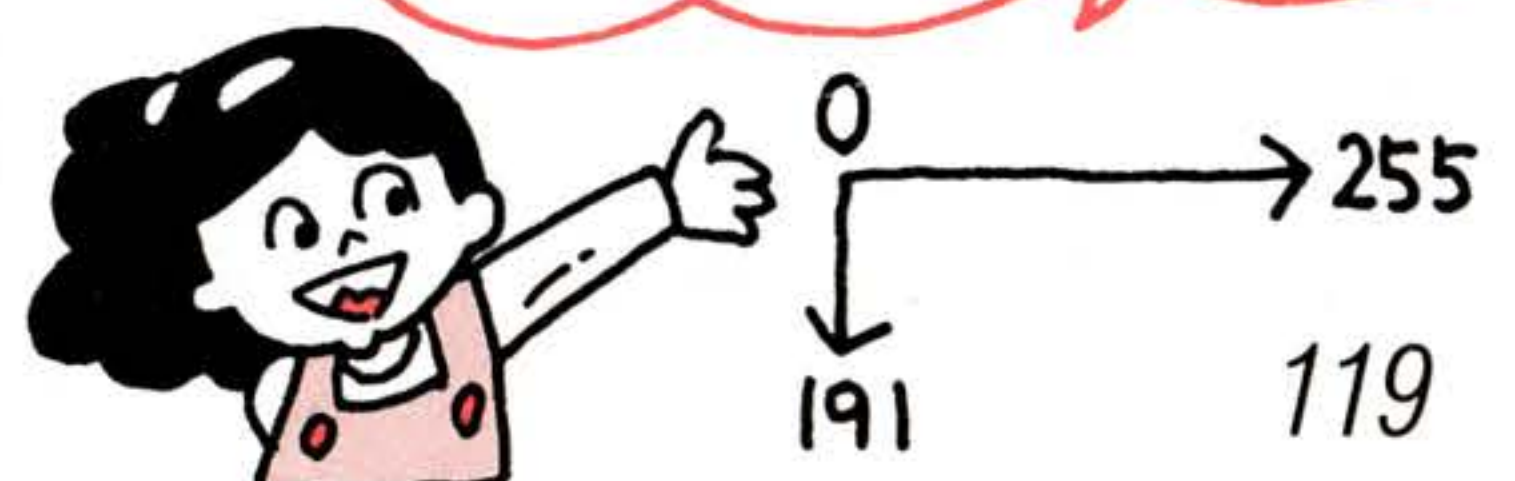


(50, 50) の位置

色は9の色(赤)

(150, 150) の位置

SCREEN 2の画面は横0~255ドット縦0~191ドットだからその範囲で指定してね





ラインめいれい LINE命令のフルコース

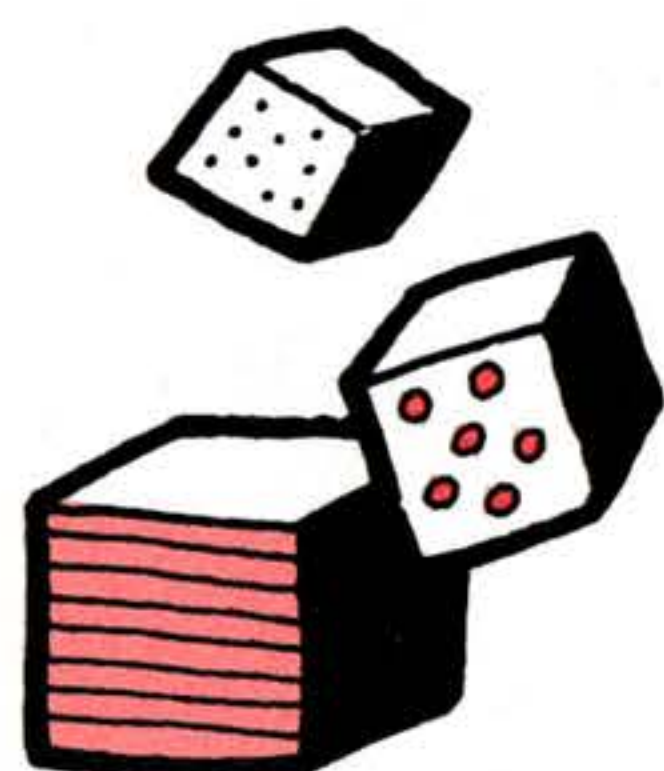
ラインめいれいせんひめいれい
LINE命令は線を引く命令だが、ただ1本の線をひくだけではなく、1行だけのラインめいれい
しかくけいはこかか
四角形の箱を描いたり、描いた箱の中を色で塗ったりもできる。



ラインぶん
LINE文は

せんひ
線を引くだけじゃない

LINE (X₁, Y₁) - (X₂, Y₂), カラーコード, BまたはBF



Bなら箱をか描き、BFなら箱を塗る

せんはこなかぬいろ
線または箱の中を塗る色

てんかんしゅうてんざひょう
2点間の終点の座標。Xは横、Yは縦

てんかんしてんざひょう
2点間の始点の座標。Xは横、Yは縦の座標

まえのページにあったラインぶん
LINE文にBをつけたして、実行してみよう。

```
10 SCREEN 2
20 LINE(50,50)-(150,150),9,B
30 GOTO 30
```

Bをつけたせば

しいてん
指定した2点を

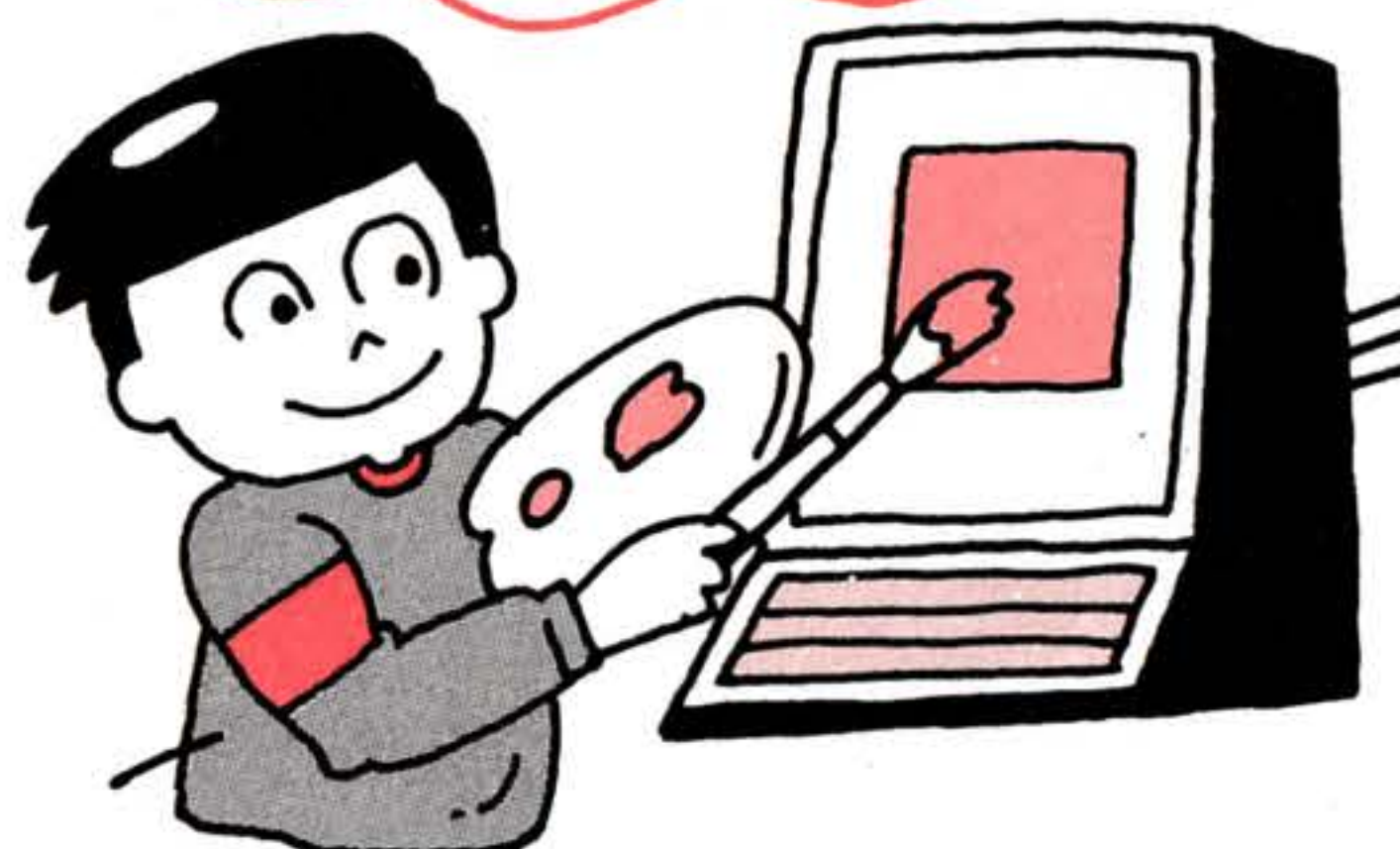
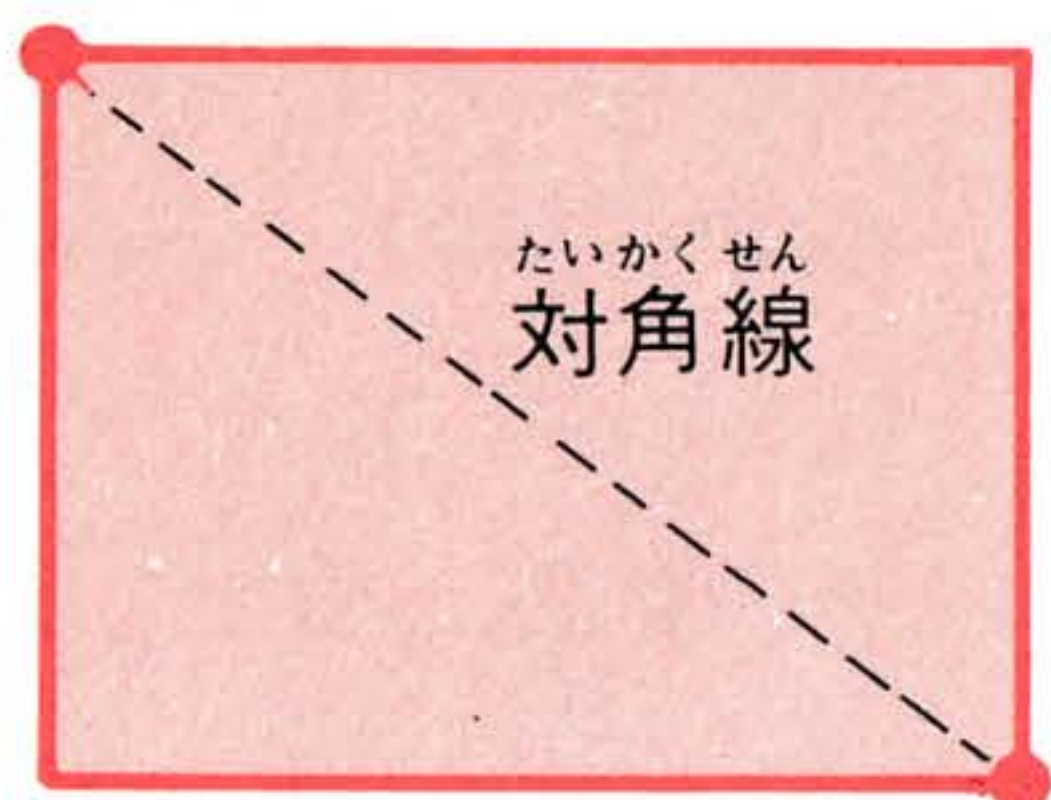
むすせん
線んだ線を

たいかくせん
対角線にした

しかくけいか
四角形を描くよ

か
BをBFに変えれば

しかくけいなかいろぬ
四角形の中を色で塗りつぶす

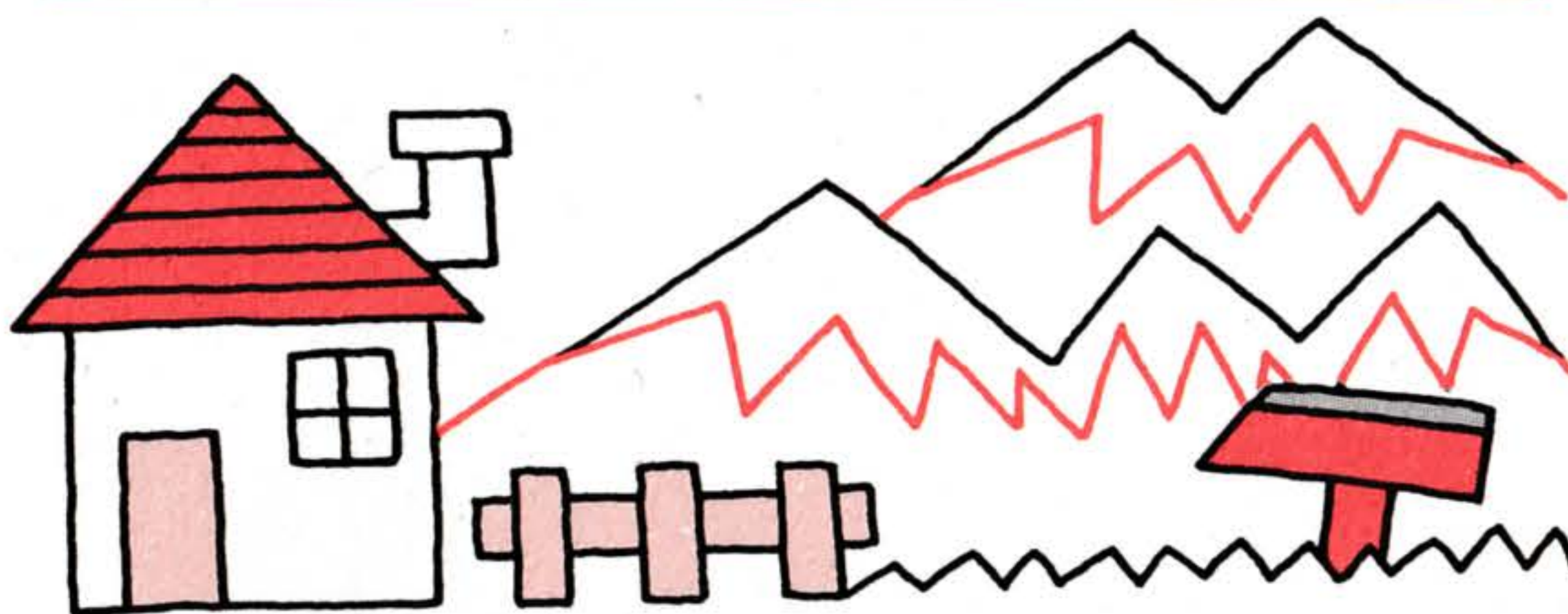
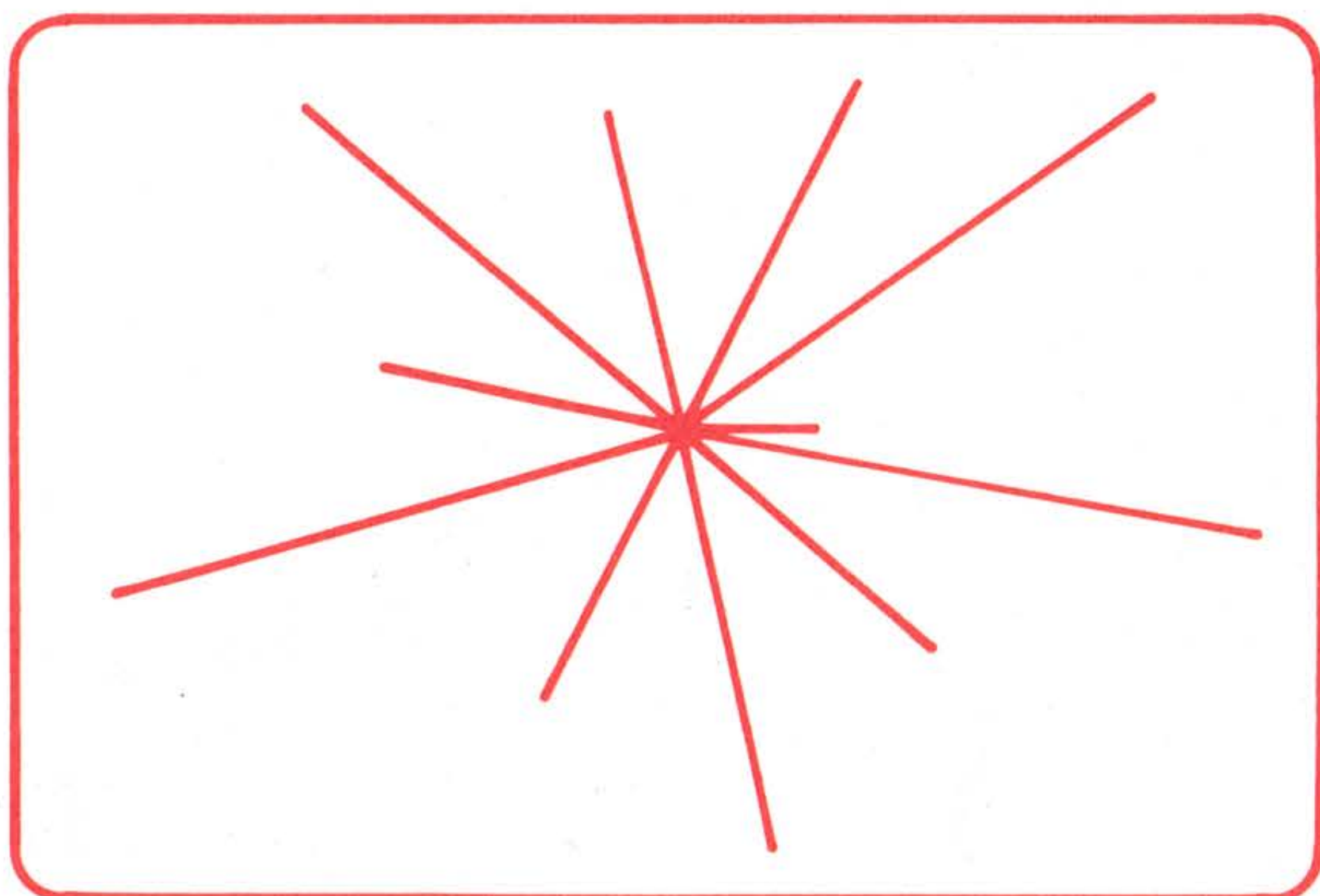


●LINE文ショートプログラム 1

●中心からいろいろな長さの線を描く

```
10 SCREEN 3
20 DEFINT A-Z
30 X=RND(1)*256
40 Y=RND(1)*192
50 C=RND(1)*15+1
60 LINE(127,95)-(X,Y),C
70 GOTO30
```

画面のほぼ中心から四方八方にいろいろな長さの線を引くプログラムだ。SCREEN 3を使っているので、ドットが小さくなっている。



注意 グラフィック命令のプログラムで、ENDで終わっていないプログラムを止めるには、**CTRL** キーと **STOP** キーを同時に押す。

■放射状に伸びる

線の基点は (127, 95) である。(直線の始点=60行)。

■各線の終点の座

標はRND関数で指定してあるからパ

ソコンまかせだ。

■SCREEN 3の

座標指定は、SCREEN 2と同じでいい。

■LINE命令で

使う (X₁, Y₁) -

(X₂, Y₂) の「-」

は、マイナスの記号である。



ごかくけい ろっかくけい か 五角形でも六角形でも描けるよ

ラインぶんしかくけいかお
LINE文で四角形を描くのは、終わりにBをつ
ければいい、と習ったね。もちろんBをつけない
で、ラインぶんほんならおな
で、LINE文を4本並べても同じだ。

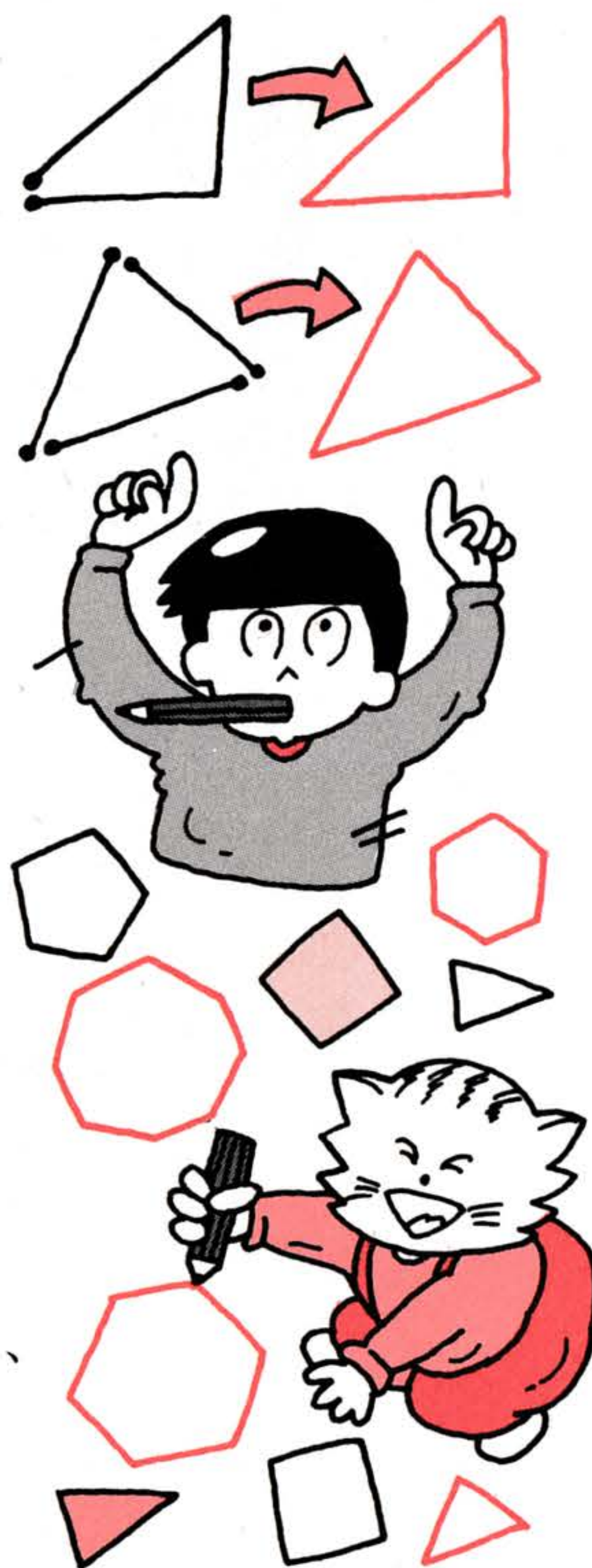
```
10 SCREEN 2
20 LINE(50,50)-(150,50)
30 LINE(50,50)-(50,150)
40 LINE(150,50)-(150,150)
50 LINE(50,150)-(150,150)
60 GOTO 60
```

うえ
上のプログラムの4本のLINE文は、

LINE (50, 50)-(150, 150), , B

おな
と同じなのだ。

おなかんがラインぶんぎょうかさん
同じように考えれば、LINE文を3行書いて三
かくけいか
角形が描けるし、5行書いて五角形が描けるのは、
かんたんにわかるよね。はっかくけいじゅうかくけい
八角形でも十角形でも、
なんでも描けるはずだ。



ステップラインぶん STEPつきLINE文

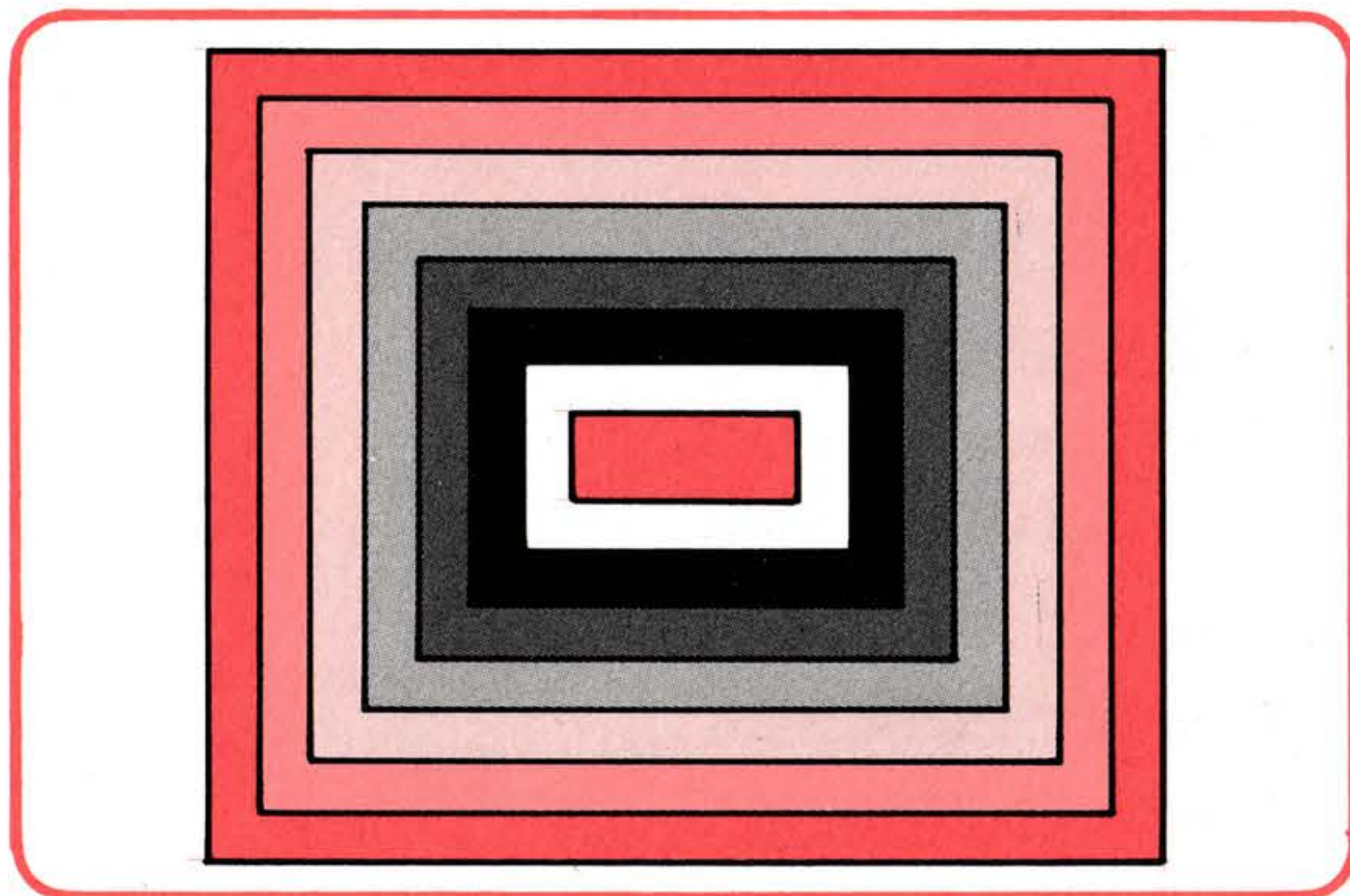
ラインぶんラインステップ
LINE文には、LINE STEP (X₁, Y₁) -STEP (X₂, Y₂) とい
ぶんけいステップ
う文型もある。STEP (X₁, Y₁) は、その直前に使った(X₁, Y₁)
ざひょうてんざひょうてん
の座標点から、(X₁, Y₁) だけずれた座標点のことだ。たとえば、
ちやくぜんつかざひょうステップ
直前に使った座標が(100, 50) で、いま、STEP (10, 10) とすれ
ば、(110, 60) の位置ということだ。

●LINE文ショートプログラム 2

●^{はこ}箱の中に^{なか}いくつも^{はこ}箱を描く^か

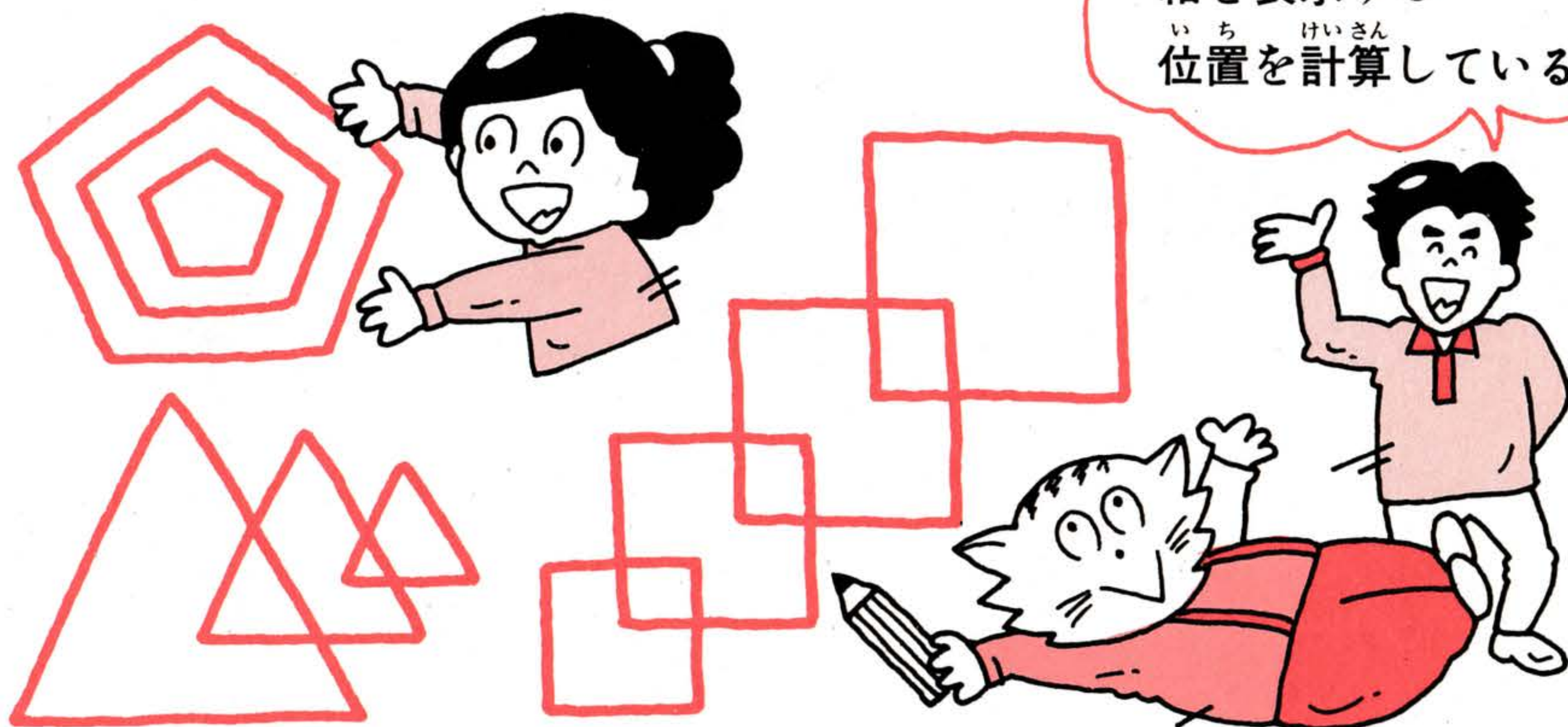
```
10 SCREEN 2
20 FOR I=1 TO15 STEP 2
30 X1=10+I*5
40 Y1=10+I*5
50 X2=240-I*5
60 Y2=180-I*5
70 LINE(X1,Y1)-(X2,Y2),I,B
80 NEXT
90 GOTO90
```

^{はこ}箱の中に^{なか}箱を描き、
またその^{はこ}箱の中に^{なか}箱
^かを描き、結局8個の
^{はこ}箱を描いてしまうプ
ログラム。^{はこ}箱を表示
する色は、^{いろ}ぜんぶ^{ちが}違
うのだ。



^{ぎょう}20行の^{へんすう}変数 I は
^{ライン}LINE命令の
^{いろ}色のカラーコードを
^{へんか}変化させているよ

^{ぎょう}30行～^{ぎょう}60行は
^{はこ}箱を表示する
^い位置を^{けいさん}計算している

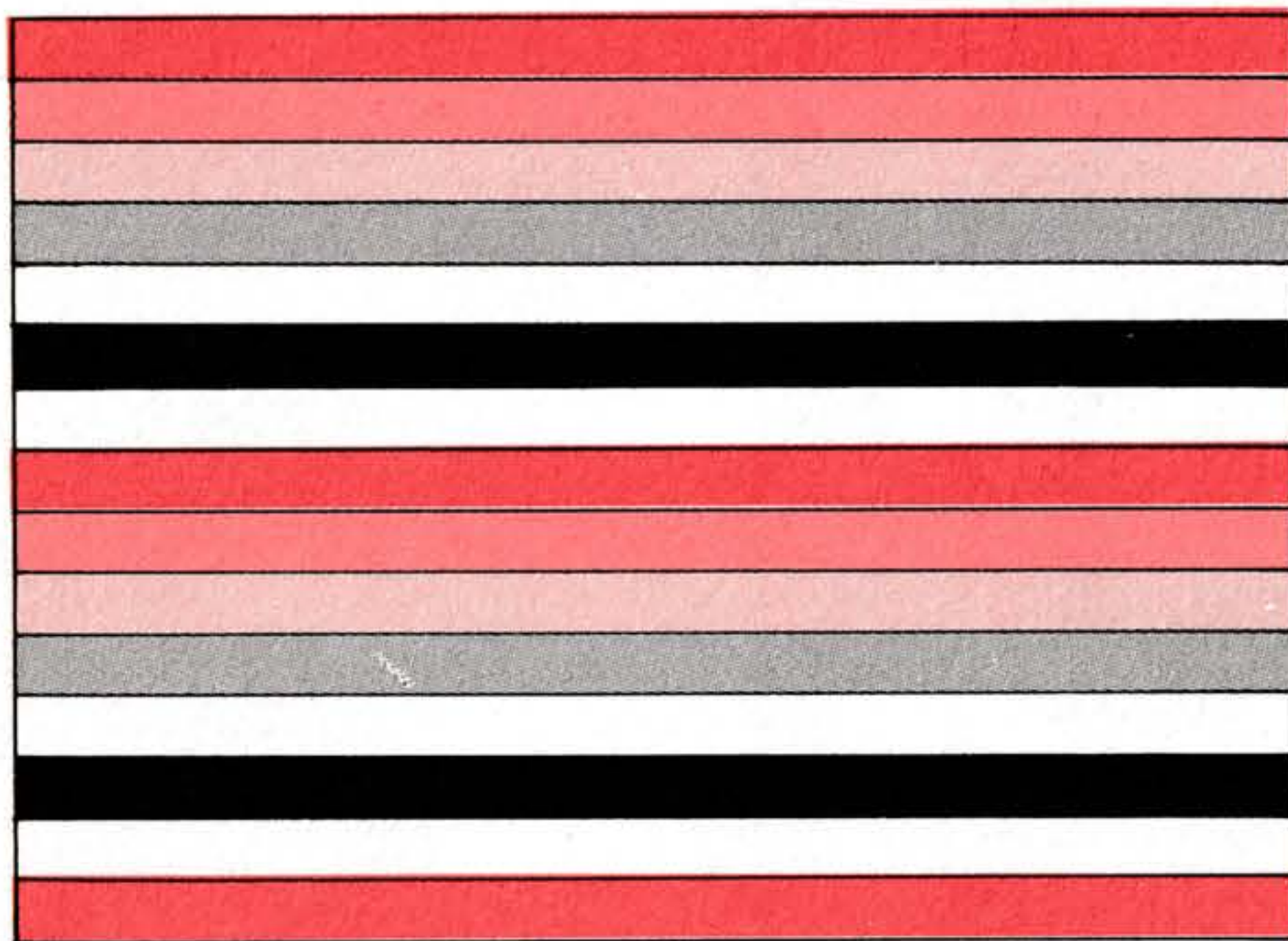


●LINE文ショートプログラム 3

●細長い箱を15個描いてそれぞれ色を塗る

```
10 SCREEN 2
20 FOR I=1 TO 15
30 Y1=(I-1)*12+5
40 Y2=Y1+11
50 LINE(10,Y1)-(240,Y2),I,BF
60 NEXT
70 GOTO 70
```

横に細長い四角
形を縦に15個並べ
て、それぞれ別の
色で表示するプロ
グラムだ。



お
終わりに

BFをつければ

箱を描いて

中を塗るのだったね

箱の中の色塗りは

LINE文に

BFをつけないで

PAINT命令で

塗っても同じだ

でも、このほうが
かんたんだよ



LINE文で色のコードを指定しないときは、50行のIはいらない。

ただし、Iの両わきのコンマ(,)は省略できない。次のようになる。

LINE (X₁, Y₁) - (X₂, Y₂), , BF



円、だ円、扇形を描く



ちゅうしん はんけい いろ
中心はどこ？半径は？どの色で？

サークル めいれい えん か
CIRCLE命令 ● だ円も描けるパソコンのコンパス

えん えん おうぎがた か サークル めいれい
円やだ円、扇形を描くのはCIRCLE命令だ。

サークル めいれい き ほんけい
CIRCLE命令の基本形はかんたんだ。

CIRCLE (X, Y), 半径, カラーコード

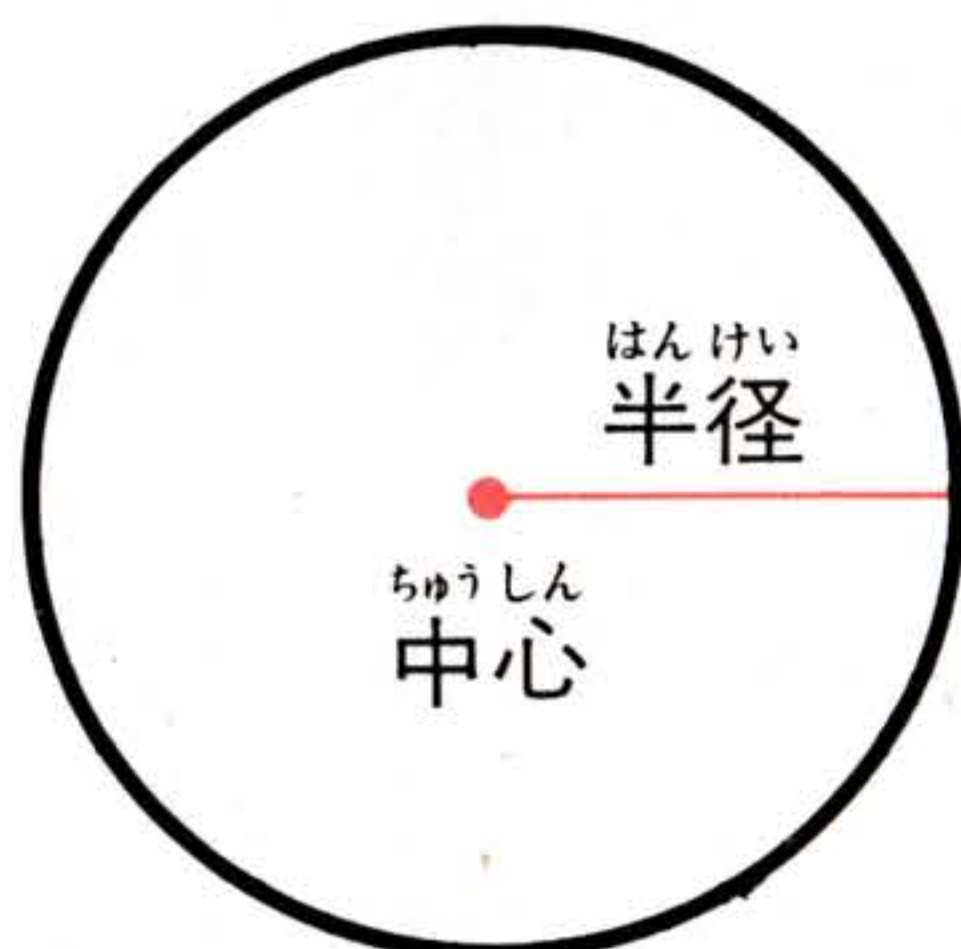
と、これでいい。

コンパスで円を描くときだって、中心の位置を
決めてから、コンパスを半径の長さに開いて、グ
ルッとまわすだろう。あれと同じだよ。

(X, Y) は中心の横の位置と縦の位置だ。半径
はドットの数だ。そして、その円をどんな色で描
くかを、カラーコードで指定してやればいい。



```
10 SCREEN 2
20 CIRCLE(130,90),50,8
30 GOTO 30
```



CIRCLE (130, 90), 50, 8 の円

カラーコード 8

あか いろ
(赤) の色

はんけい
半径は50ドット

ちゅうしん い ち よこ たて い ち
中心位置は横130, 縦90の位置



かい し かく ど しゅうりょうかく ど くわ えん こ 開始角度と終了角度を加えれば円弧

CIRCLE文はコンパスだから、ぐるりとまわって円を描いてしまう。その円周上を、ここからここまでしか描かなくていいよ、と命令できれば、円弧（アーチ）ができるはずだ。

円を描く基本形を思い出してみよう。

CIRCLE (X, Y), 半径, カラーコード
だったけれど、これに「開始角度」と「終了角度」というものを加えてみる。

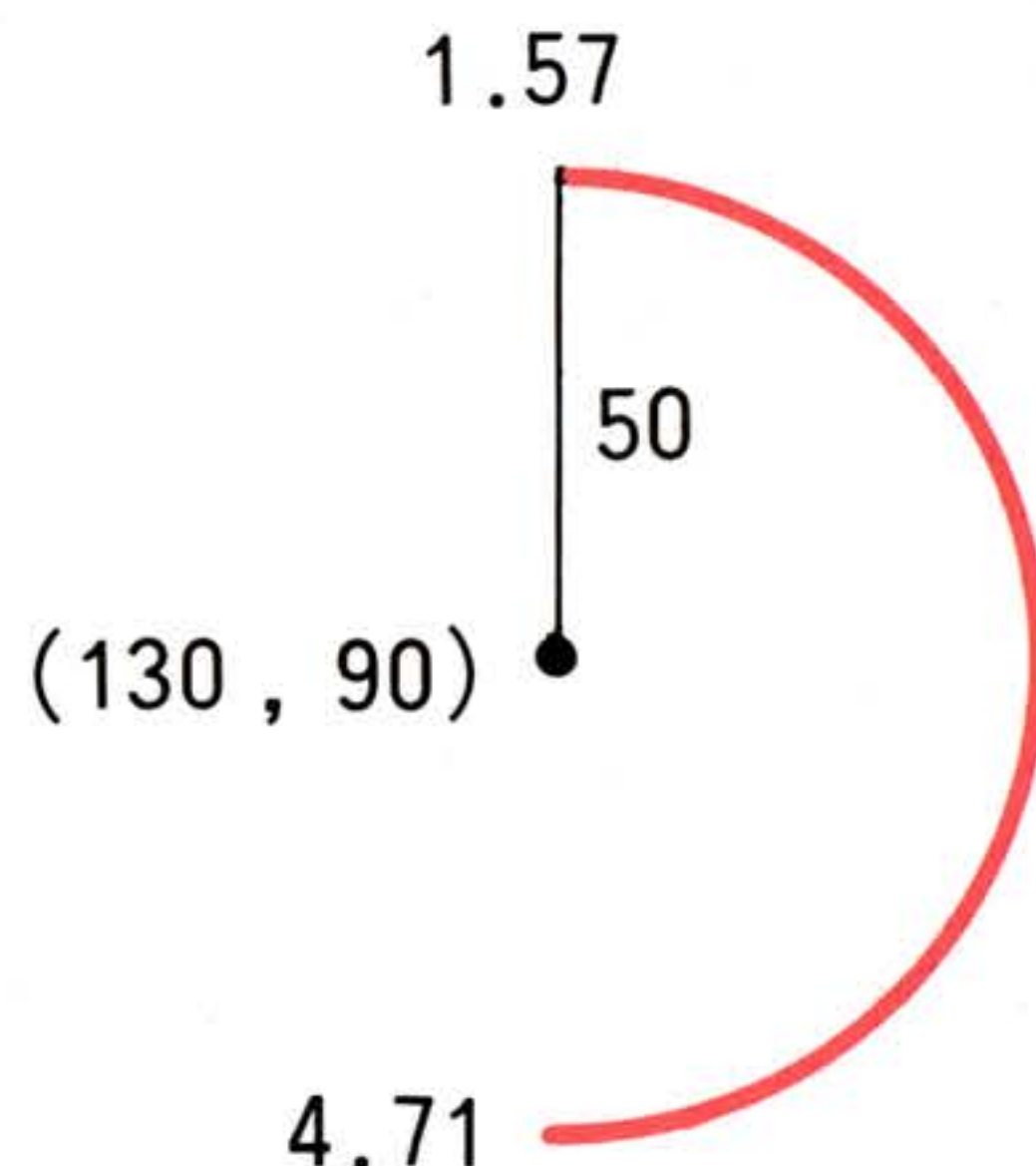
CIRCLE (X, Y), 半径, カラーコード,
開始角度, 終了角度

ということになる。

これが円弧を描くCIRCLE文だ。

```
10 SCREEN 2
20 CIRCLE(130,90),50,8,4.71,
  1.57
30 GOTO 30
```

画面にはどんな図が描かれたかな？



角度はラジアンで指定するのだが、むずかしいからほかの方法でおぼえたほうがかんたんだよ



ラジアンけいさんの計算

ラジアンを正確に計算する公式は
角度 $\times \pi / 180$

だ。0°から180°の半円を描く終了角度は、

$$180^\circ \times 3.14 \div 180$$

$$= 3.14 \text{ ラジアン}$$

180°が3.14だから、90°なら半分の

1.57というように、

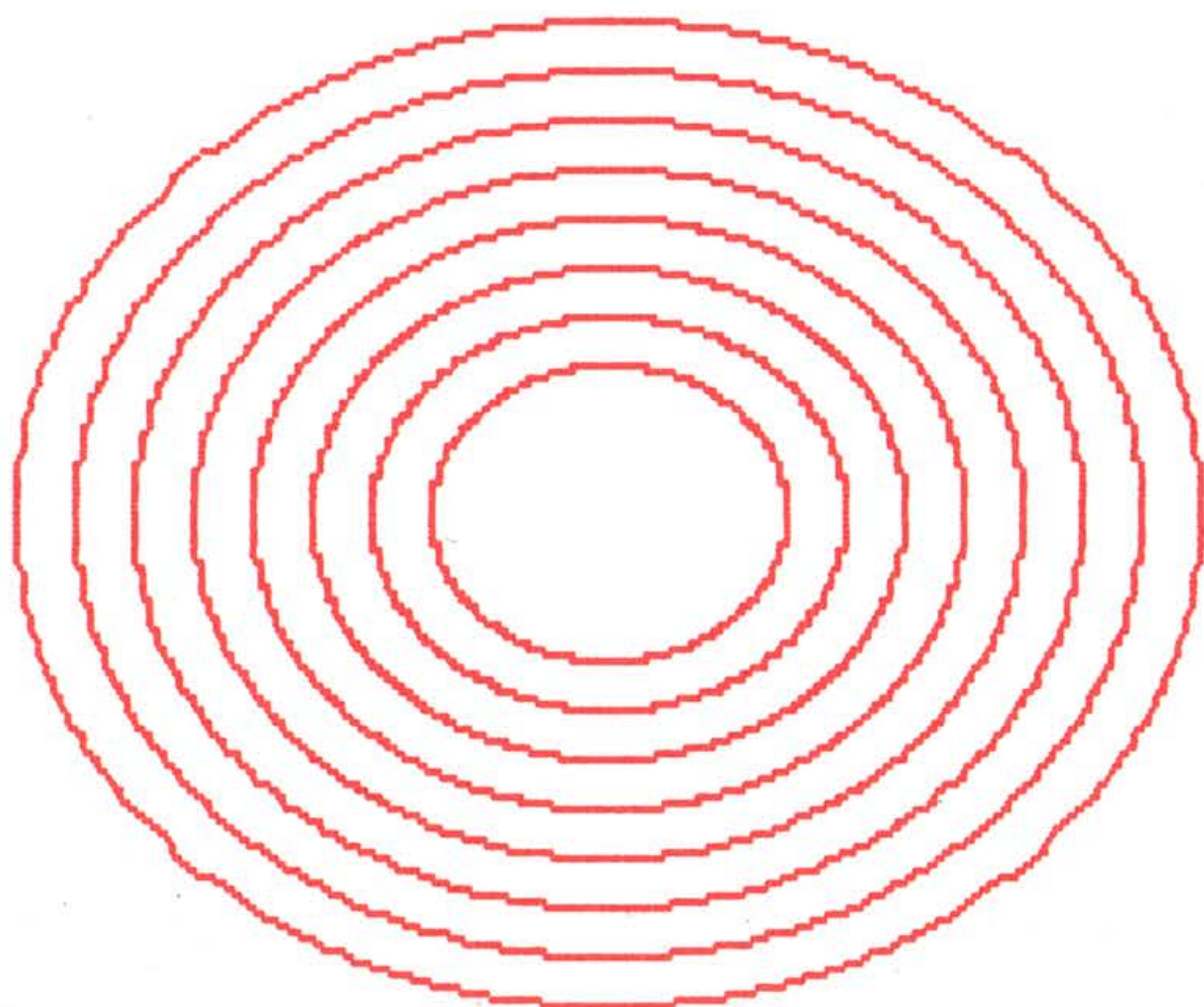
だいたいの数を暗

算してもいい。

●CIRCLE文ショートプログラム 1

●画面に8個の円を8色使って描く

```
10 SCREEN 2
20 FOR I=1 TO15 STEP2
30 R=I*4+20
40 CIRCLE(120,90),R,I
50 NEXT
60 GOTO 60
```

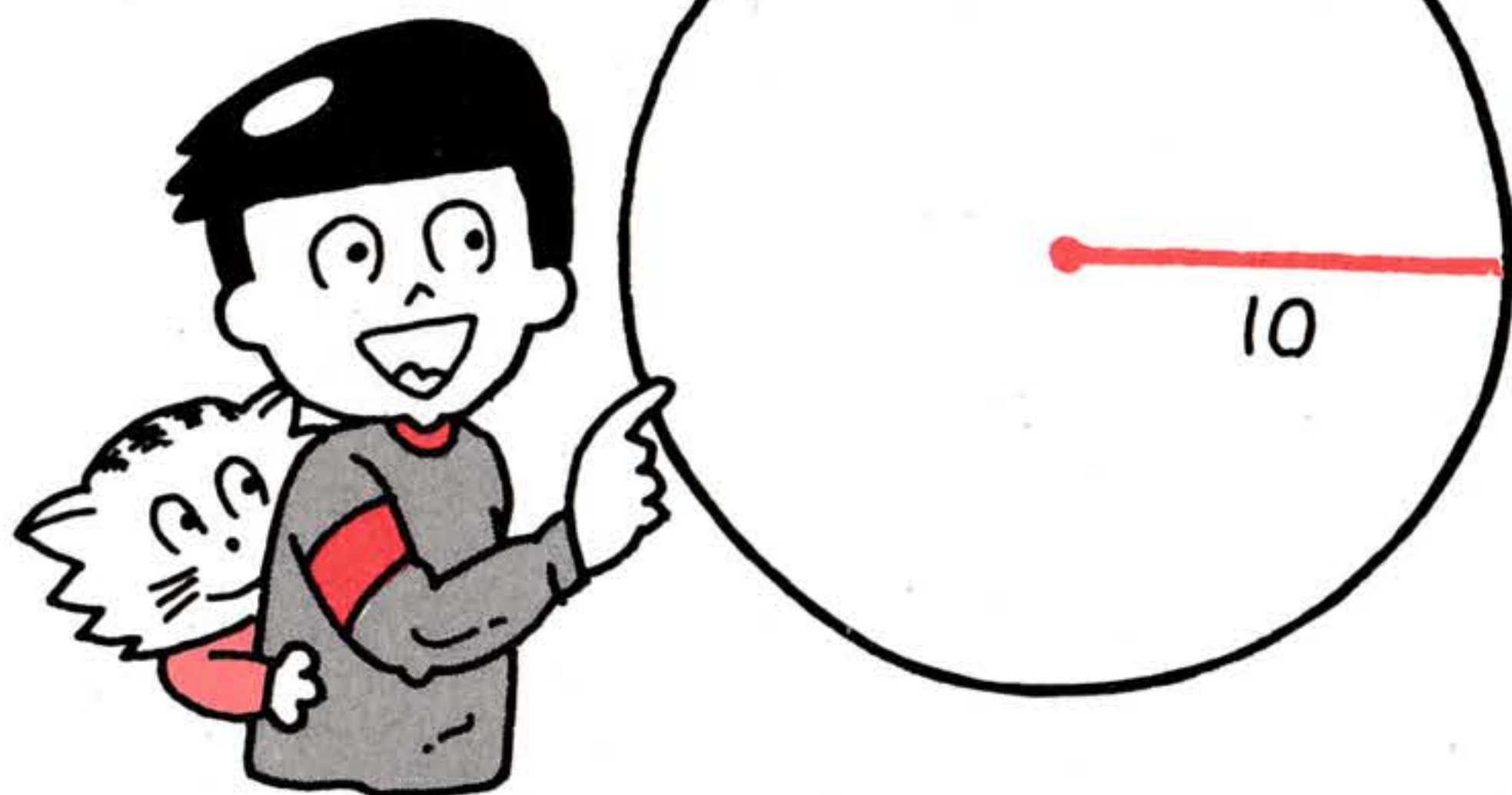


20行のFOR文は、描く色の指定をしている。カラーコード1～15までの色を、STEP 2で順に描いていく。

だから、カラーコード1、3、5、7、9、11、13、15の8色だ。

左の図はプリンタで出力したからだ円になっているけどディスプレイにはちゃんと円で表示されるよ

この円の半径は10だ



円の半径はX軸方向のドットの数だよ

40行のCIRCLE文で、Rは半径だ。30行で計算した8通りの値が半径になる。

なぜ8通りかということ、もう一度20行を見よう。20行のIは8通りあるから、40行のIも同じく8通りあると考えようよ。



かくど おうぎがた 角度にマイナスをつければ扇形だ

サークルぶん きほんけい かいしかくど しゅうりょうかくど
CIRCLE文の基本形に開始角度と終了角度をつ

けたして、アーチ（円弧）を描く方法をおぼえた
けれど、これを応用すれば、扇形も描けるはずだ、
とは考えられないかい？

アーチを描いて、その線の端と円の中心を線で
結べば、扇形ができるのは、だれにもわかるよね。

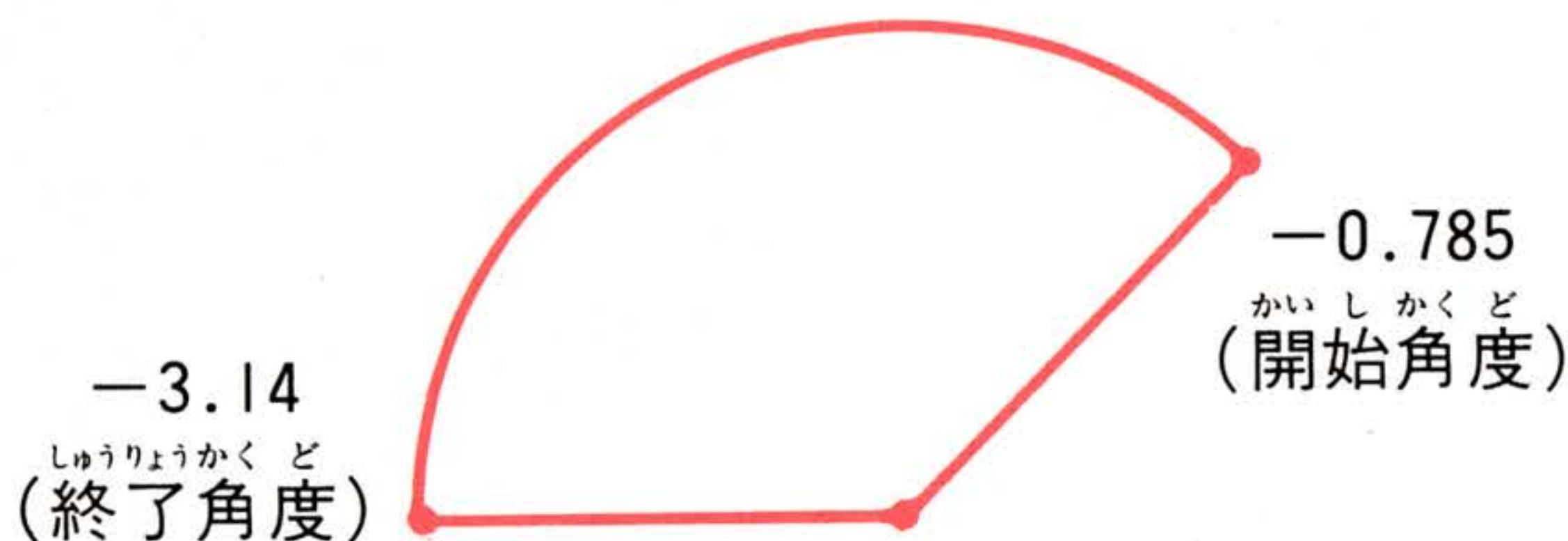
もったいぶらないで、教えちゃおう。アーチを
描くときの開始角度と終了角度にマイナス（-）
をつければ、円弧のできあがり。

126ページの扇形を描くプログラムを少し変え
て、マイナスをつけてみよう。

CIRCLE (X, Y), 半径, カラーコード,
-開始角度, -終了角度

だから、次のプログラムになるね。

```
10 SCREEN 2
20 CIRCLE(130,90),50,8,-.785,
  -3.14
30 GOTO 30
```



かくど
角度（ラジアン）
の指定は
-2π（-6.28）～
2π（6.28）の範囲
でないと
イリーガル ファンクション
Illegal function
コール
call のエラーに
なるよ



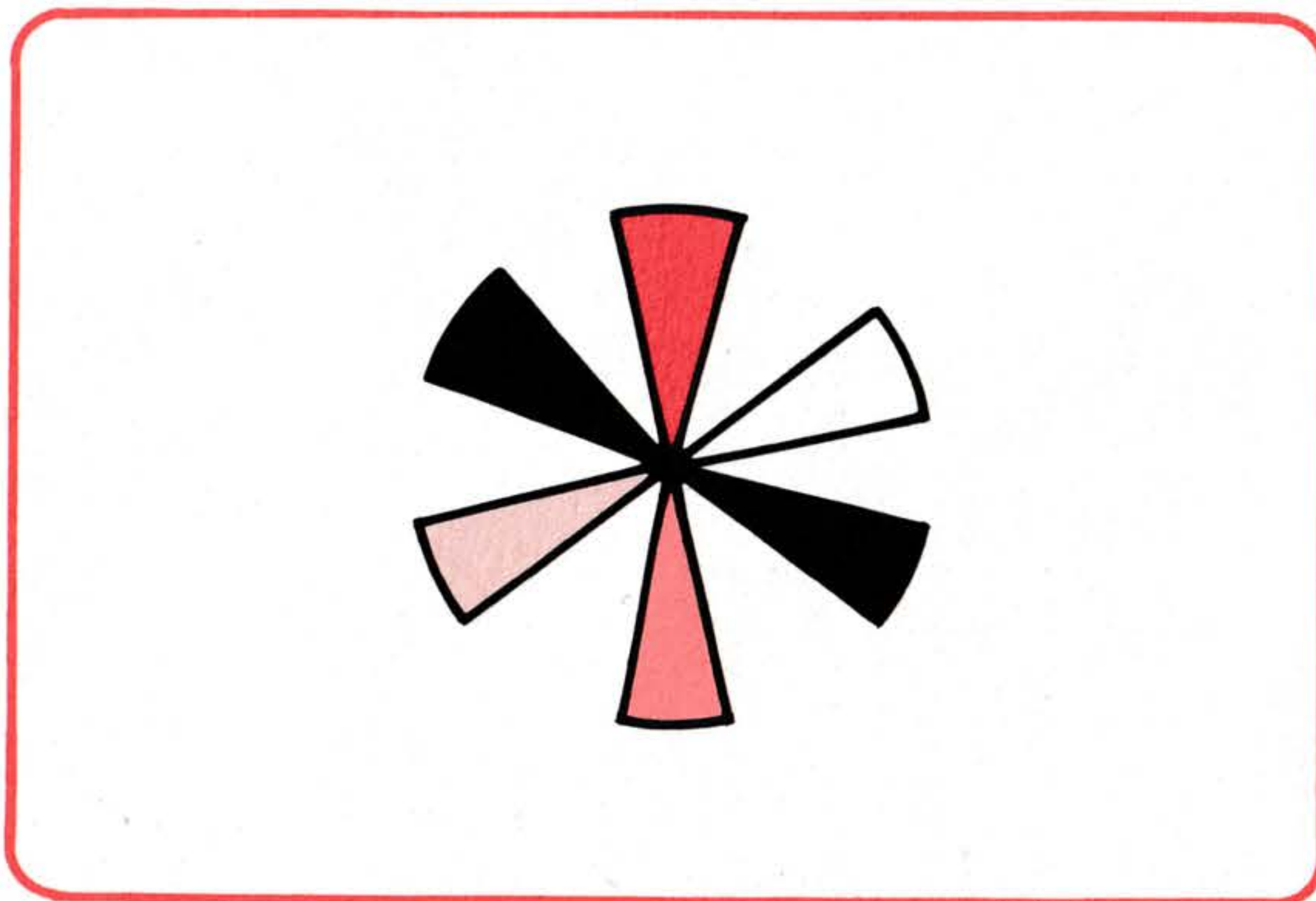
●CIRCLE文ショートプログラム 2

●放射線状に6個の扇形を描く

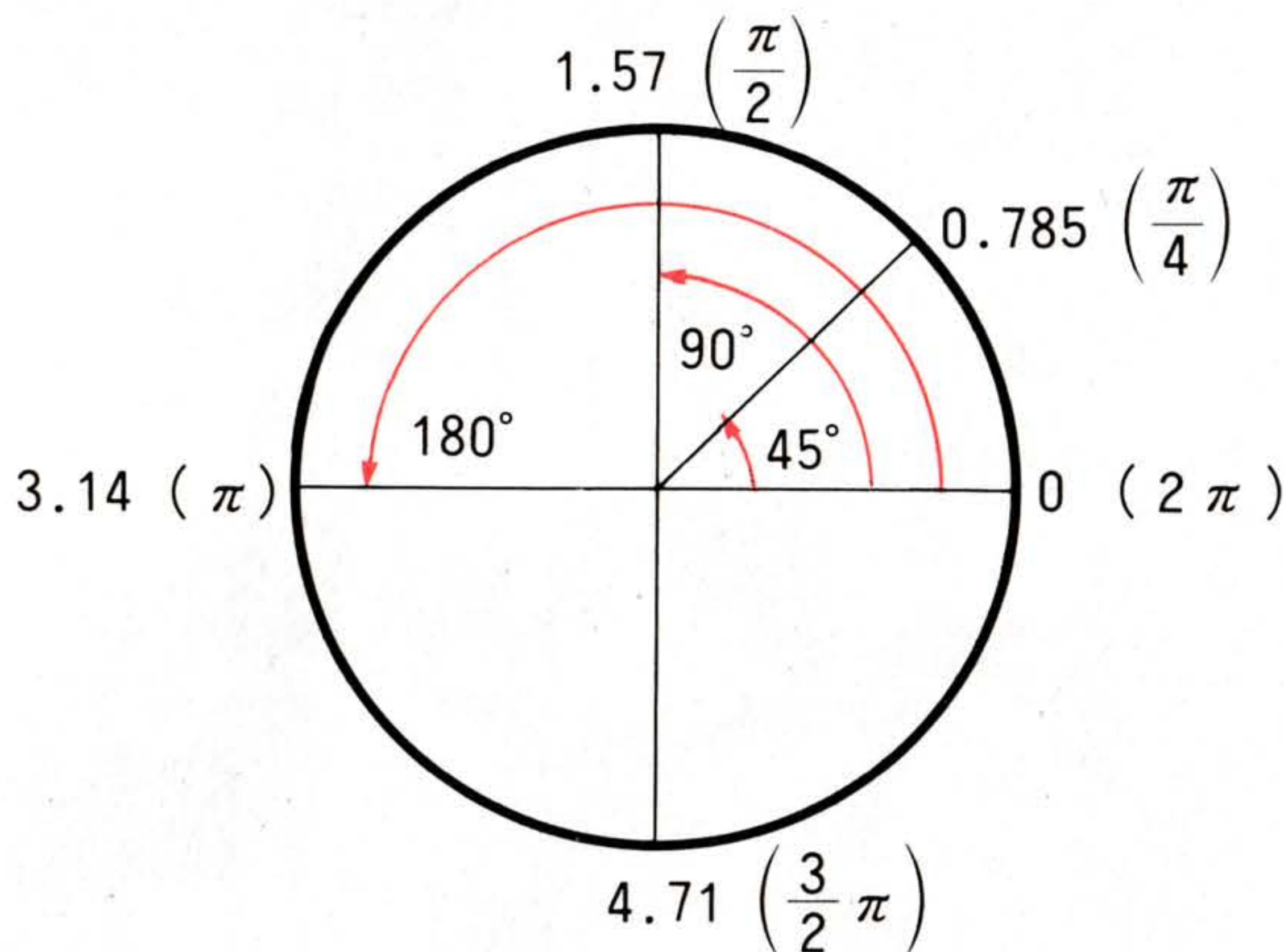
```

10 SCREEN 2
20 FOR I=0 TO 5
30 R1=I*3.14/3+3.14/12
40 R2=R1+3.14/6
50 CN=I*2+1
60 CIRCLE (120,90),70,CN,-R1,-R2
70 NEXT
80 GOTO 80
    
```

扇形を6個
それぞれ違う
色で描くプ
ログラムだ。
マイナスをつ
け忘れないで。



●角度を指定するラジアン値



60行のCIRCLE
文で、表示する色
はCN、開始角度
はR1、終了角度
はR2と変数にな
っている。
扇形を描くのだ
から、R1、R2
にはマイナス(-)
をつけてある。

20行のFOR文で
変数Iは0～5の
6種類。だから50
行の色の指定はカ
ラーコード1、3、
5、7、9、11と
いうことだね。



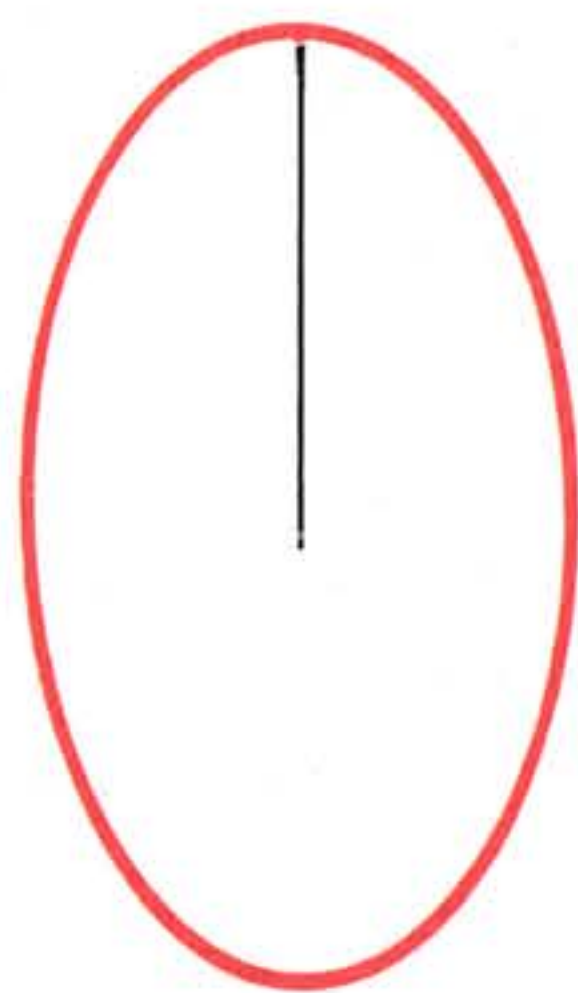
縦横の比率を与えてだ円を描く

円を描いて、アーチ（円弧）を描いて、扇形を描いたら、その次はだ円形を描いてみよう。円やだ円を描くには、開始角度と終了角度はいらないよね。だから、そこは省略して、最後に「比率」というものをつけたす。

CIRCLE (X, Y), 半径, カラーコード, , , 半径

実際にやってみよう。

```
10 SCREEN 2
20 CIRCLE(130,90),50,8,,,2
30 GOTO 30
```



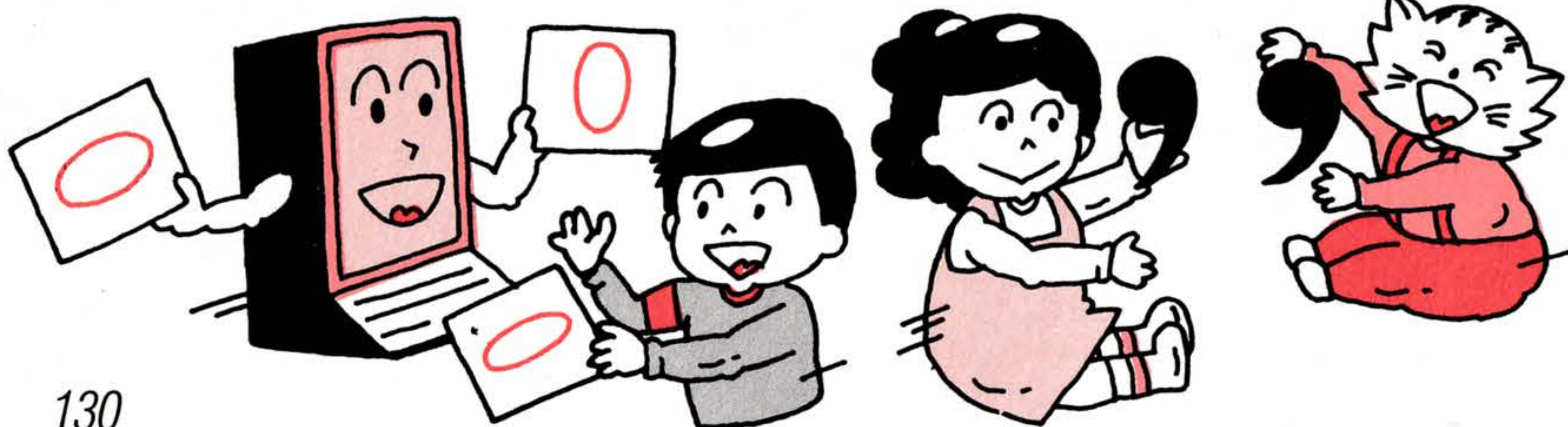
比率は (垂直方向半径) ÷ (水平方向半径)

比率が1なら
真円。省略してもいい

比率が1より大きければ縦長だ円

比率が1以下なら横長だ円

開始角度と終了角度を省略しても
2つのコンマ(, ,)は省略できない

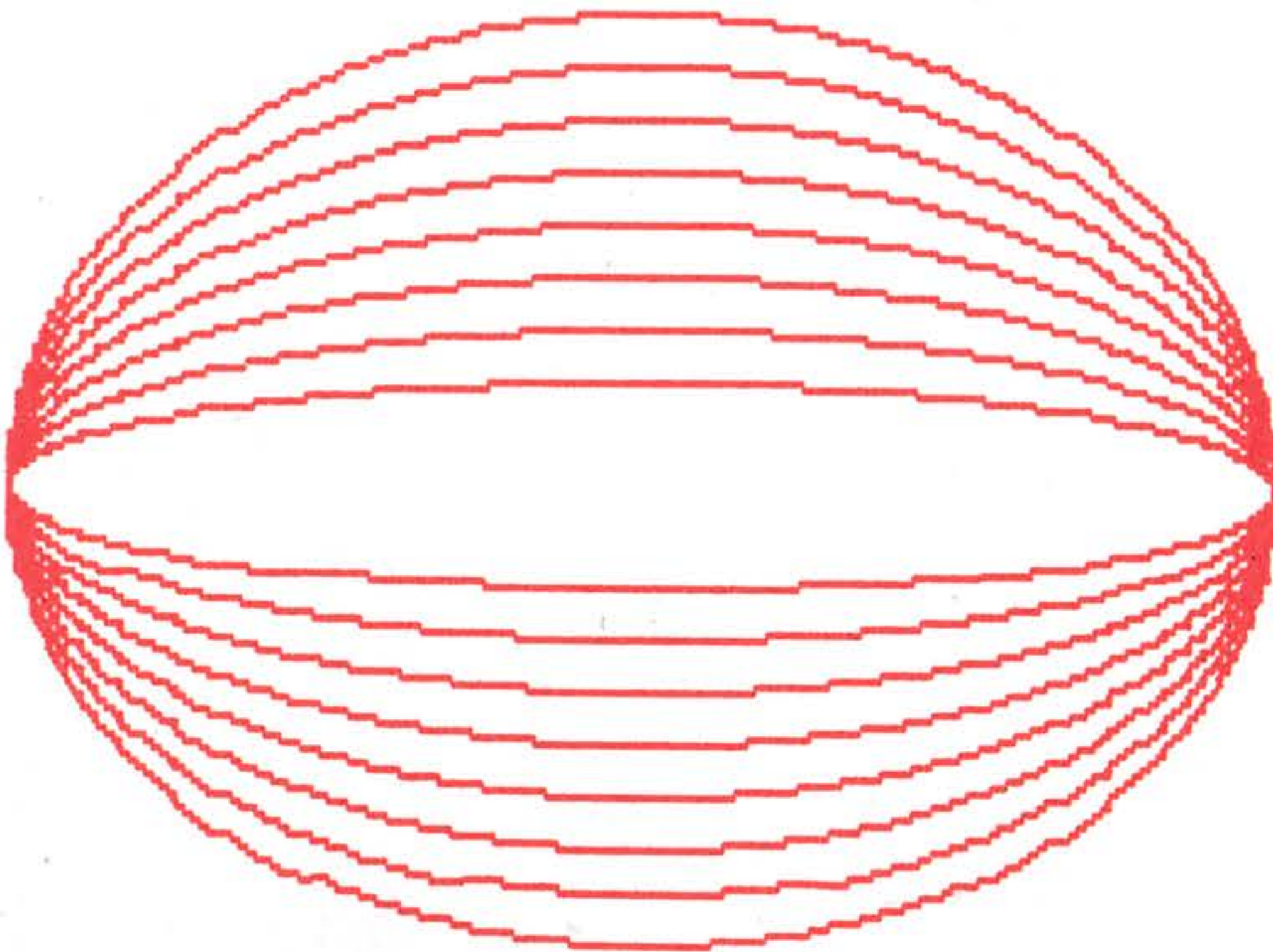


●CIRCLE文ショートプログラム 3

●横長のだ円形を8個描く

```

10 SCREEN 2
20 H=1
30 FOR I=1 TO 15 STEP 2
40 H=H-.1
50 CIRCLE (120,90),80,I,,H
60 NEXT
70 GOTO 70
    
```



50行のCIRCLE

文で

Hの前にコンマが

「, , ,」と

3つ並んでいるが

これは

開始角度と終了角

度を

省略したシルシ

40行の「.1」は

「0.1」のことだよ



50行のCIRCLE文で、

比率を変数Hにしている。

20行でH=1と定義し、

40行でHを0.9~0.2まで

ひとつずつ小さくしてい

る。H=H-.1をH=H

+.1にすれば、縦長のだ

円になるよ。

4 5

線で囲んだ領域に 色を塗る



でぐち ずけい か
まず出口のない図形を描いてから

ペイント めいれい きょうかいしよく かこ ぼしよ りょういきしよく ぬ
PAINT命令●境界色で囲まれた場所を領域色で塗る

がめん かんぜん せん かこ ずけい か
画面に完全に線で囲んだ図形を描いたとする。

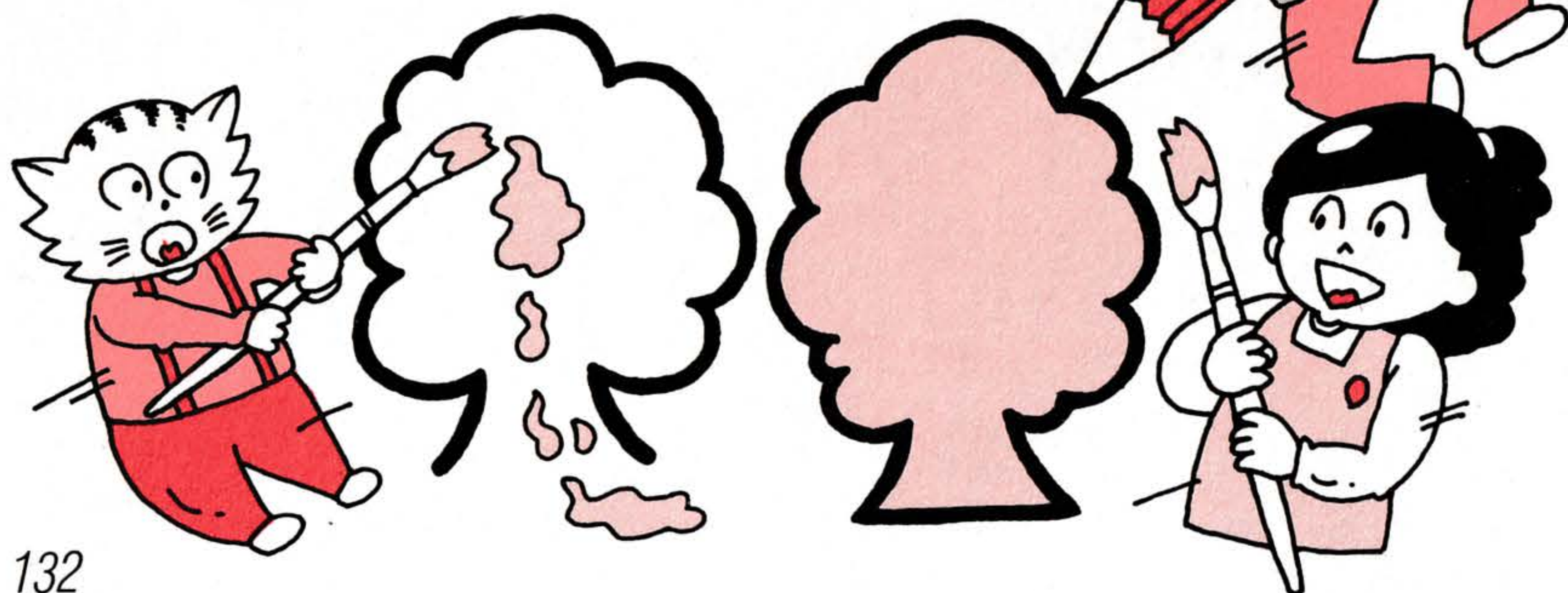
○でも△でも□でも、どんな図形でもいいのだが、
たとえば、□や∪のように出口があいている図形
はいけない。完全に囲まれた部分に、べったりと
色を塗るのがPAINT命令だ。

せいかく してい きょうかいしよく かこ ぼしよ
正確にいうと、指定した境界色で囲んだ場所を
指定した領域色で塗るということだ。

○なら○の図形で、○の図形の線の色を境界色、
○の中に塗る色を領域色というわけだ。

○の中に色インクをたらして塗っても、インク
は外に流れ出さないが、□や∪は出口があいてい
るから、塗ったインクが外に流れ出してしまう。

ラインぶん
LINE文や
サークルぶん
CIRCLE文で
でぐち
出口のない
ず か
図を描いて
それから
ペイントぶん
PAINT文だ





図形内にある場所を指定して

図形の中に色を塗るPAINT文は、

PAINT (X, Y), 領域色, 境界色

というように使う。

(X, Y) は例によって画面の場所を指定する。

Xは横の位置、Yは縦の位置だ。そして、色を塗りたい図形の中なら、どこかの場所を指定してもいい。

領域色は図形の外わくの色。カラーコードで指定するんだよ。

境界色が、べったり塗りたいインクの色だ。やっぱりカラーコードで指定するんだ。

領域色と境界色

スクリーン
SCREEN 2

境界色は使用で

きない。だから、

境界色も領域色も

指定した領域色で

塗られるから、外

わくはつかない。

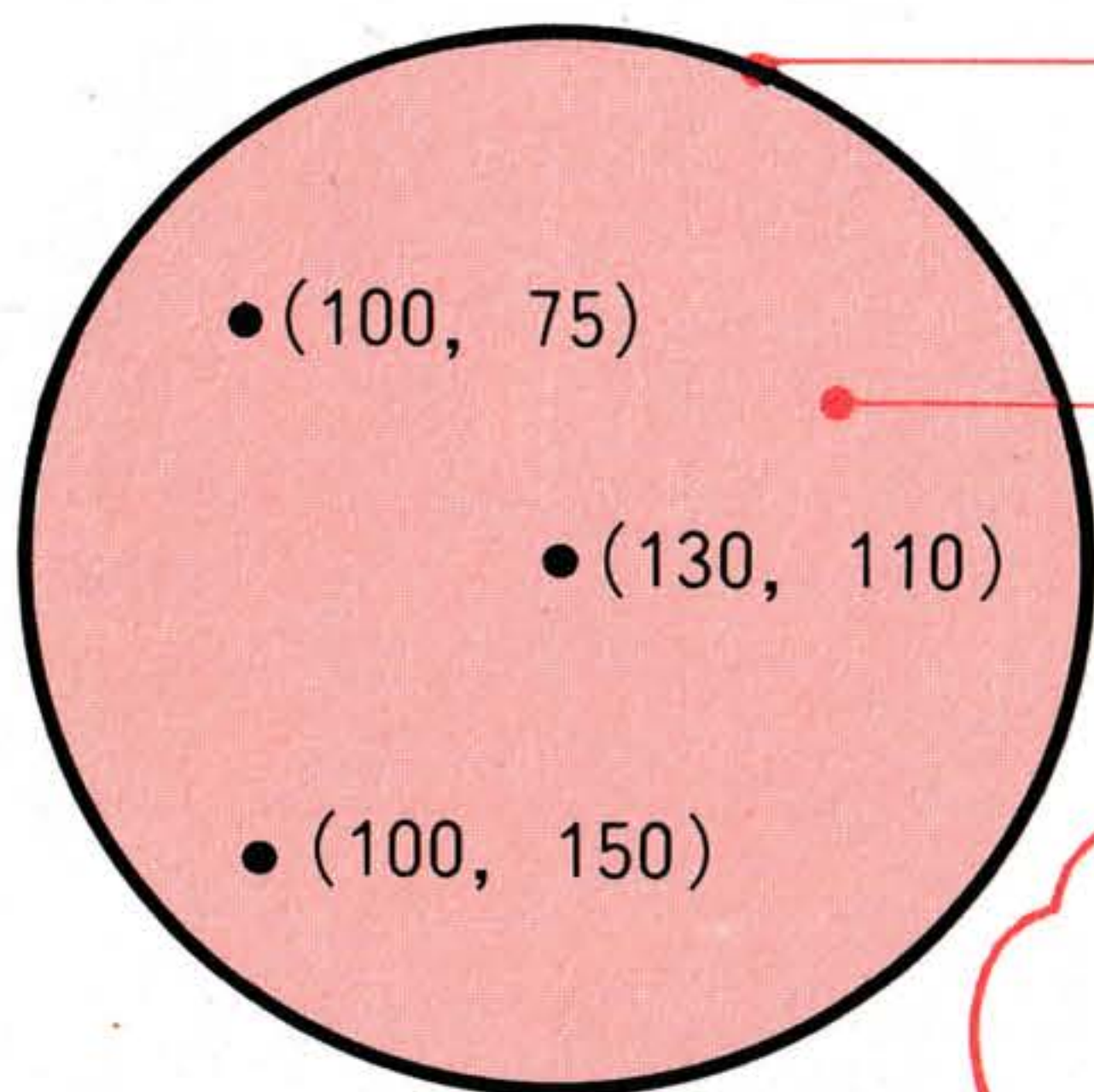
スクリーン
SCREEN 3

領域色、境界色

とも使えるから、

外わくと内部を別

の色で塗れる。



境界色：図形の外わくの色

領域色：境界色で囲まれた図形の内部の色



スクリーン
SCREEN 2のときは

外わくを別の色では塗れないよ

領域色を省略すると：COLOR文で指定した前景色になる。

境界色を省略すると：領域色と同じ色になる。



スクリーン スクリーン SCREEN2とSCREEN3

かんたんなプログラムを実行してみよう。

```
10 SCREEN 3
20 CIRCLE(130,100),50,8
30 PAINT(130,100),2,8
40 GOTO 40
```

カラーコード8の赤い外わく、内部が2の緑色で塗られた画面が表示されただろう。

こんどは、10行のSCREEN 3を2に変えて、CIRCLE文の最後の8を2に変えて実行してみよう。外わくはなし、ぜんぶ緑色で塗られた円が表示されたはずだ。

ペイント文の
PAINT文の

(130, 100) は
円の内部なら
どの位置の
座標でもいい



スクリーン
SCREEN 3の画面

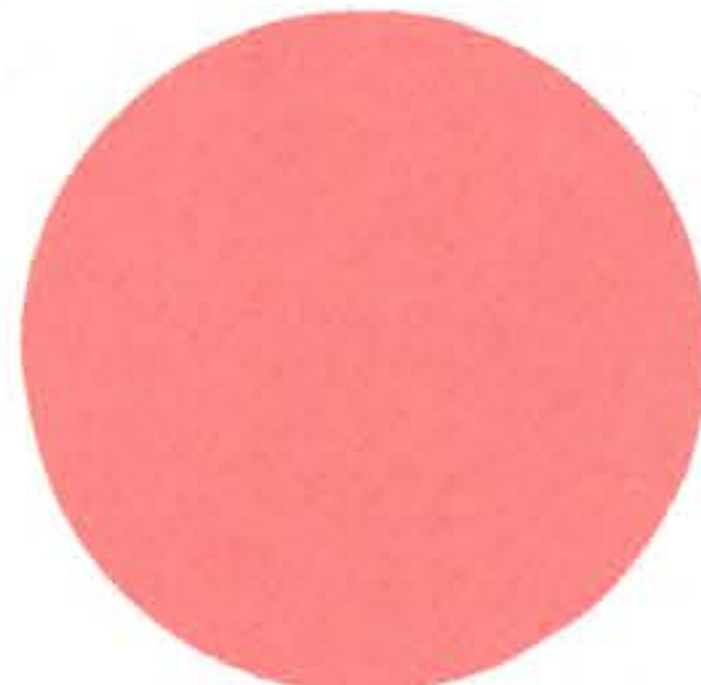
スクリーン
SCREEN 2の画面

境界色は赤



領域色は
緑色

境界色も領域色も緑色



スクリーン 2 横256×縦192ドットの高解像度グラフィックモード。

スクリーン 3 より目がこまかい。

スクリーン 3 横64ブロック×縦48ブロックのマルチカラーモード。

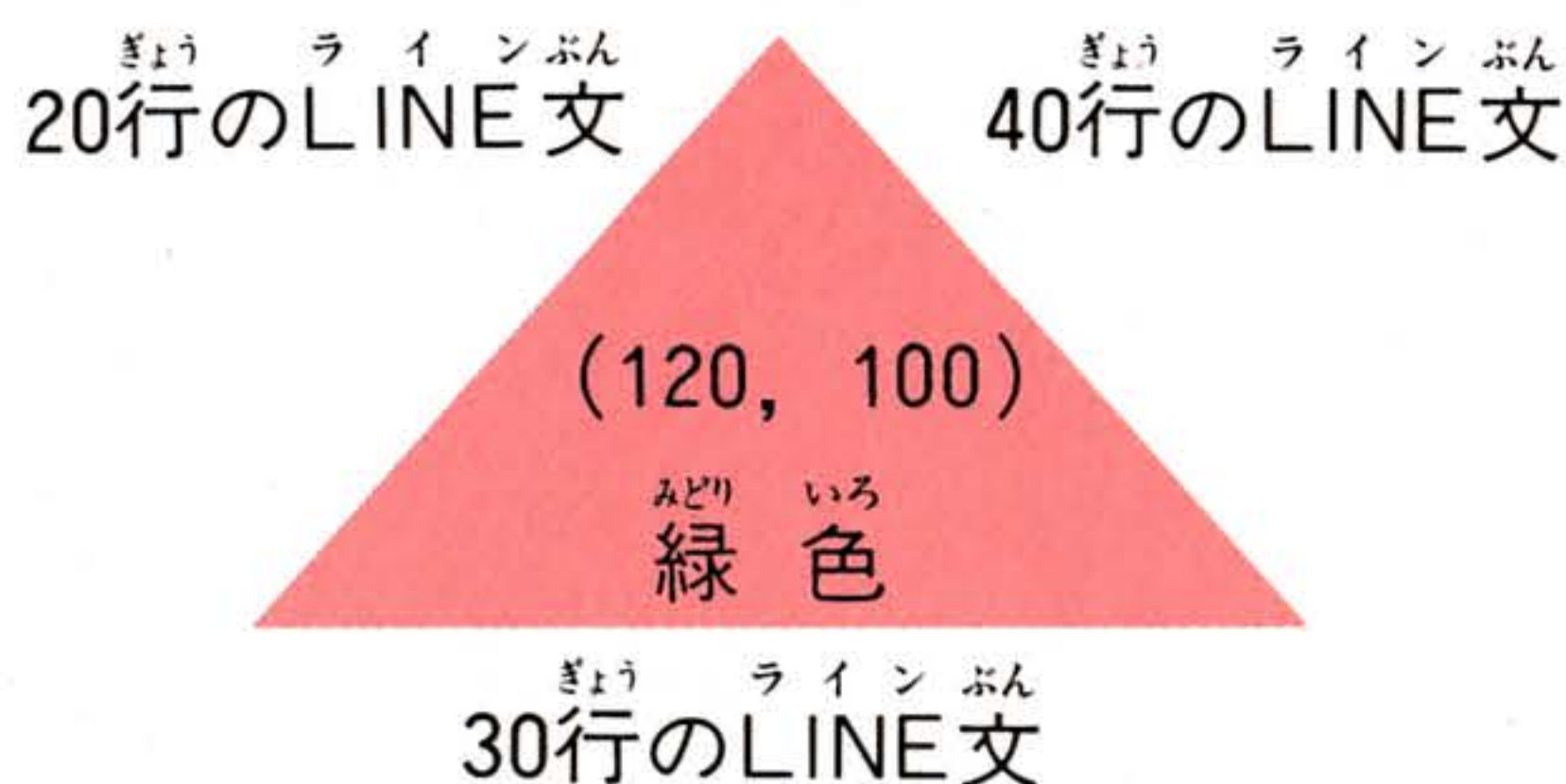
ただし、座標の指定は256×192ドットと同じ。

●PAINT文ショートプログラム

●^{さんかくけい}^{なか}^ぬ三角形の中を塗りつぶす

```
10 SCREEN 2
20 LINE(120,60)-(40,140),2
30 LINE-(200,140),2
40 LINE-(120,60),2
50 PAINT(120,100),2
60 GOTO 60
```

①



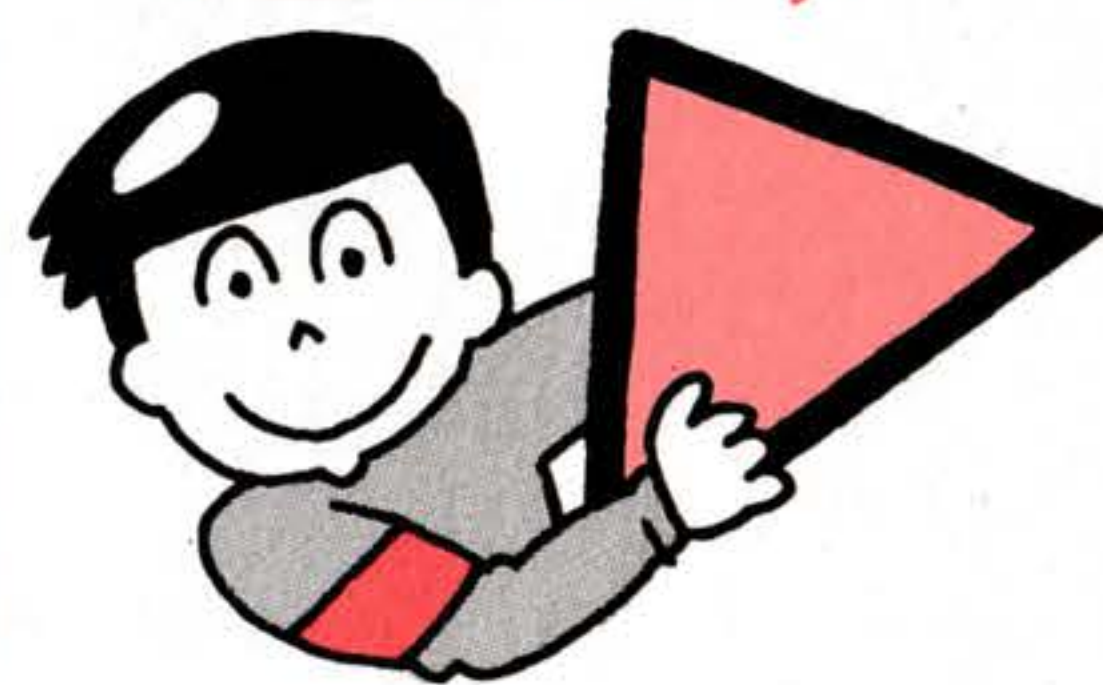
^{ライン}^{ぶん}^{ほん}^{かく} LINE文 3本で三角
^{けい}^か^{うちがわ} 形を描き、その内側を
^{みどりいろ}^ぬ 緑色で塗りつぶす。

^{スクリーン} SCREEN 2だから、
^{ライン}^{ぶん}^し^{てい}^{せん} LINE文で指定した線
^{いろ}^{おな}^{いろ} の色と同じ色（カラー
コード 2、^{みどりいろ}^{りょう} 緑色）の領
^{いきしよく}^し^{てい} 域色を指定している。

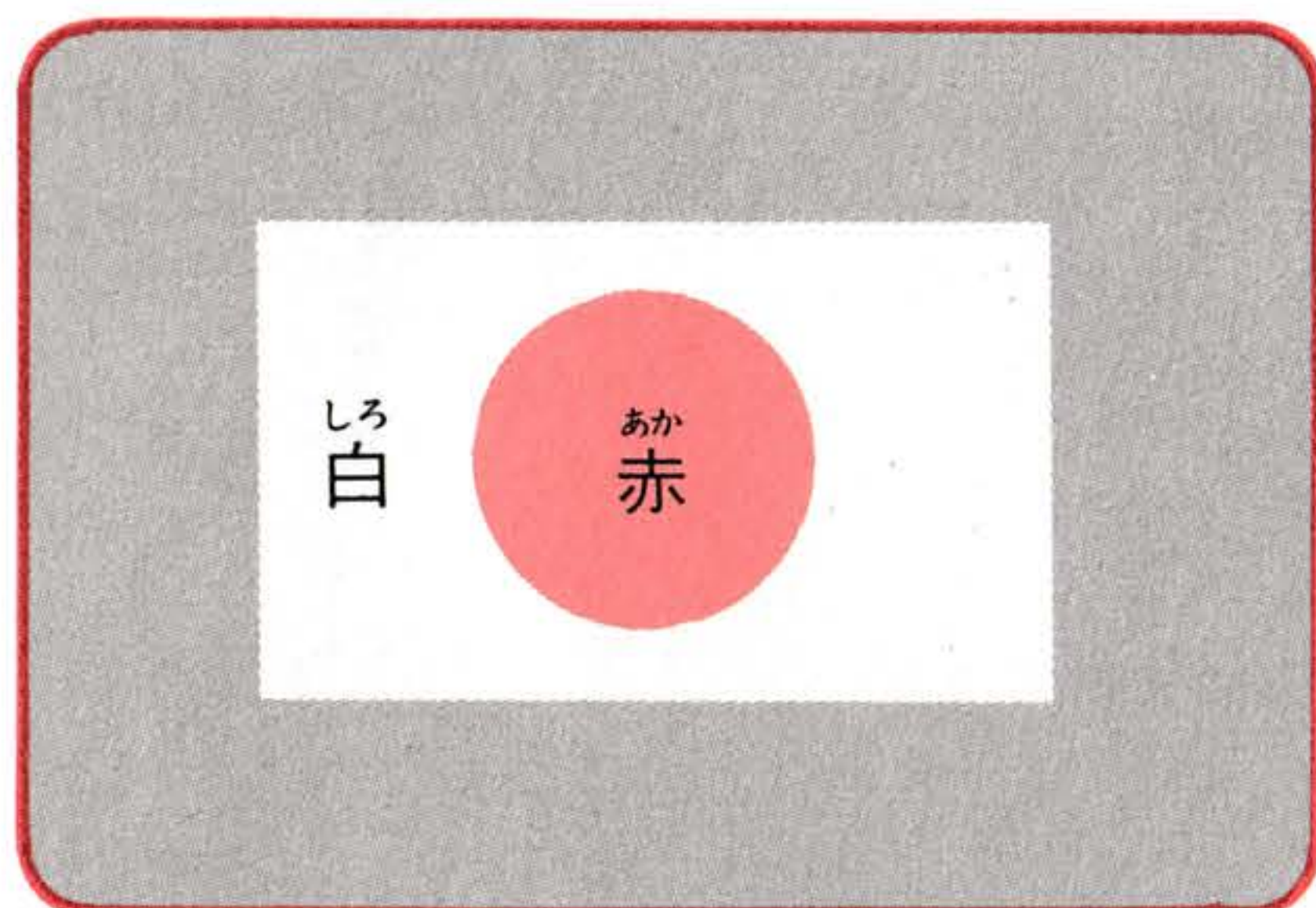
●^ひ^{まる}^{ひょうじ}日の丸を表示する

```
10 SCREEN 2
20 LINE(50,50)-(200,150),15,BF
30 CIRCLE(125,100),35,8
40 PAINT(125,100),8
50 GOTO 50
```

②



^{スクリーン} SCREEN 3なら
^{なか}^ぬ^{いろ}^{べつ}^{いろ} 中を塗る色と別な色で
^{そと}^ぬ 外わくを塗れるよ



^ひ^{まる}^{はた}^{ひょうじ} 日の丸の旗を表示するプログ
ラム。ここまでですでおぼえ
てしまったグラフィック命令の
^{ライン}^{ぶん}^{サークル}^{ぶん}^{ペイント} LINE文、CIRCLE文、PAINT
^{ぶん} 文が、みんな入っているから復
^{しゅう} 習だ。

4

6

ドットに色をつけて 表示させる



どの位置のドットを表示させるか

POINTSET 命令 ● 指定した座標の点を表示

グラフィック画面には、SCREEN 2 と、SCREEN 3 の 2 通りがあるのだったね。そして、SCREEN 2 は横256、縦192のドット（点）がうめこまれているし、SCREEN 3 は横64、縦48のブロックに分かれているということだった。

このドットやブロックを指定して、あそのドットを表示させなさい、ここのブロックを表示させなさいというのが、POINTSET 命令なのだ。それも、お好みの色で表示させることができるのだ。

PSET (X, Y), カラーコード

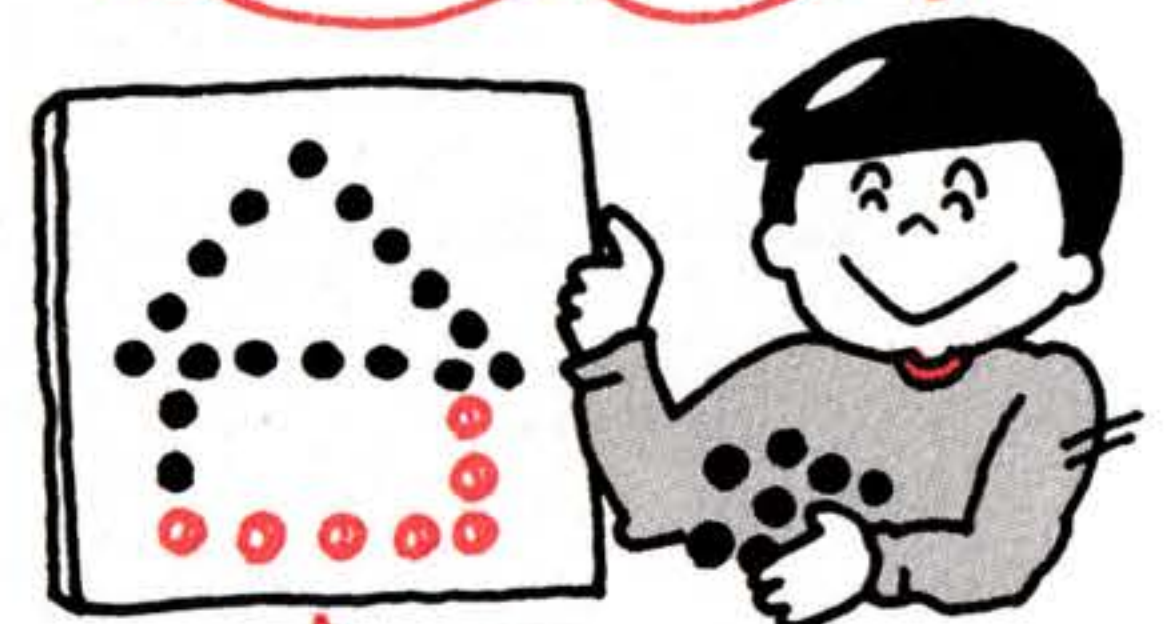
(X, Y)の位置のドットまたはブロックを、カラーコードの色で表示させるというわけ。

```
10 SCREEN 2
20 PSET (120, 90), 8
30 GOTO 30
```

画面の中央あたりに、小さな小さな赤い点が表示されたはずだよ。点が小さいので、赤い点には見えないかもしれないが。

ドットやブロックは
絵を表示させる

もとだから
画素というんだよ



座標 (120, 90)
の位置に

カラーコード 8 の
赤い点を表示した

赤い点

●PSETショートプログラム

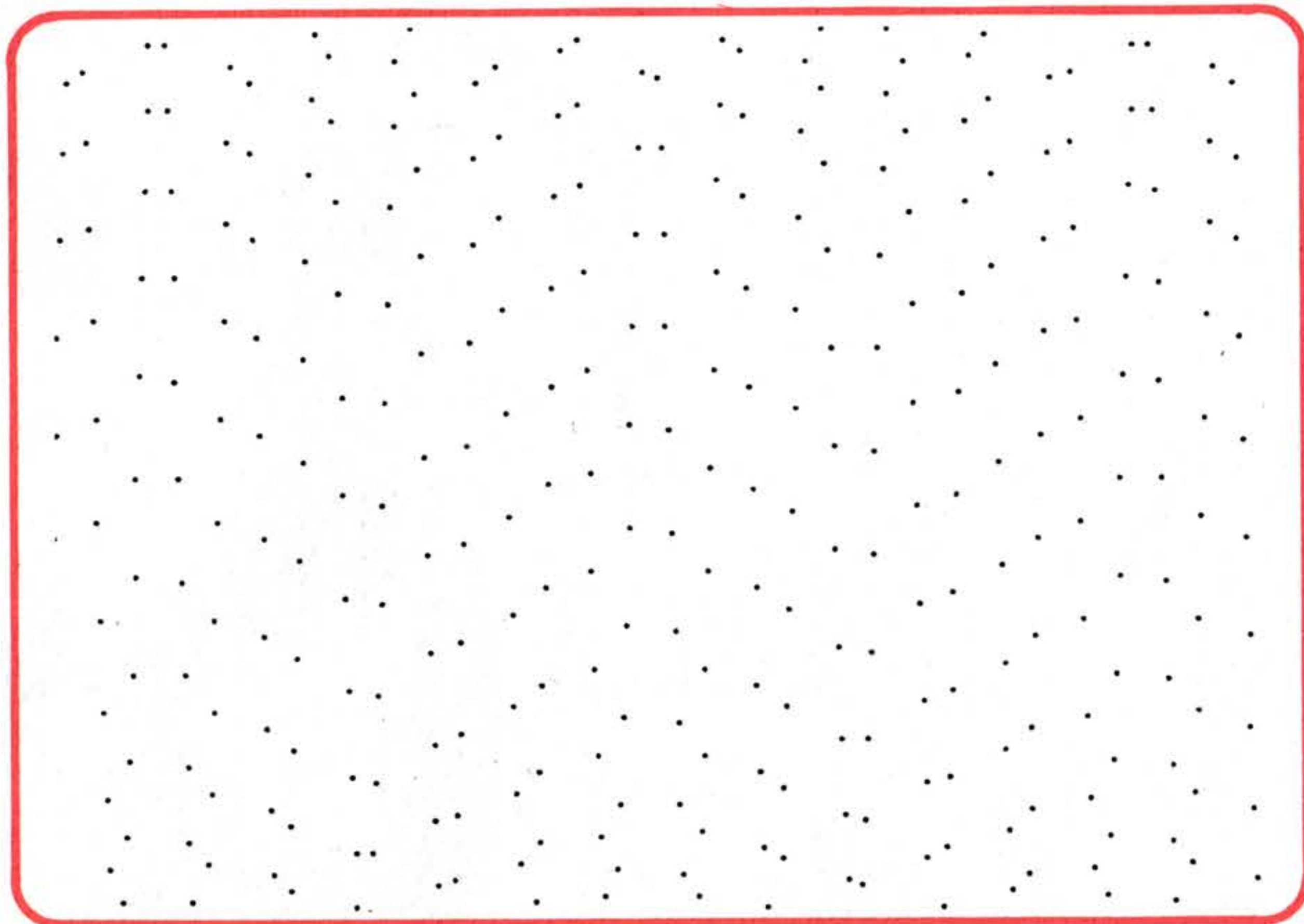
●よく見れば^みSIN^{サイン}カーブの^{てんてんてん}点点点……

```

10 COLOR 15,1
20 SCREEN 2
30 CN=2
40 FOR X=1 TO255 STEP .4
50 Y=90*SIN(X/2)
60 IF 8/X=INT(8/X) THEN CN=CN+1
70 IF CN=16 THEN CN=2
80 PSET(X,Y+100),CN
90 NEXT
100 GOTO100
    
```

ステップ
STEP .4は
ステップ
STEP 0.4の
ことだよ

ポイントセット
PSETは
かくれたドットを
ひっぱり出して
目で見えるように
するんだな



40行は横方向（X軸）の値を1から255まで変化させている。STEPのあとの数字を小さくするほど、模様はこまかくなるよ。

50行は、SIN関数を使って縦方向（Y軸）の値を計算している。

60行、70行は色を決めている。ちょっと複雑な計算だから、いまはわからなくても気にしない。



4

7

点を消す方法で 点を表示させる方法



つか かた ポイントセット おな
使い方しだいでPSETと同じ

ポイントリセットめいれい してい いち てん け
PRESET命令 ● 指定した位置の点を消す

てん ひょうじ ポイントセット ひょうじ
点を表示させるのがPSETなら、表示されてい
る点を消すのがPRESETだ。

PRESET (X, Y), カラーコード

(X, Y)で指定した位置にある点(ドット)を
カラーコードの色で消すという働きをする。

ところが実際には、消したはずの点が、表示さ
れたようにみえる使い方があるからおもしろい。

```
10 SCREEN 3
20 LINE(20,10)-(100,50),8,BF
30 PRESET(60,30)
40 GOTO40
```

がめん あかいろ はこ ひょうじ ちゅうおう
画面に赤色の箱が表示されたあと、その中央に

くら みずいろ ひょうじ おも
暗い水色が表示されたと思うよ。なぜ？



き どうじ はいけいしょく みずいろ
起動時の背景色は水色。そ
うえ あか め はこ
の上に赤で塗った箱がある。

あか ポイントリセット け
その赤をPRESETで消すか
ら、下地の水色が表面に出て
きた。とこうかんが
考えれば、わか
るだろう。

●PRESETショートプログラム

●どんどん点を消していく

```
10 SCREEN 2
20 LINE(50,50)-(200,150),2,BF
30 FOR I=50 TO150
40 FOR J=50 TO200
50 PRESET(J,I)
60 NEXT J,I
70 END
```

20行のLINE文
で描いた緑色の長
方形。50行にある
PRESET文は、四
角の緑色を上から
順番にはぎとって
いく。

ここから1ドットずつ
緑色が消されていく

緑色

緑色をはぎとら
れた部分は、下に
塗ってある色（水
色）が表面に出て
新しい点を表示し
ていくようだ。



起動時の背景色は
カラーコード7の水色
だから
上に重ね塗りした色を
PRESETでとれば
下地の水色が現れるよ



4

8

パソコンのペン描き



ほうこう なが おも
方向も長さも思いのまま

ド ロ ウ めいれい が めん じ ゆ う ず け い か
DRAW命令 ● 画面に自由な図形を描く

ディスプレイの中に絵の具とペンがかくれている、それを使って自由に絵を描くのが、DRAW命令だ。ペンを思ったとおりに動かせるから、どんな絵でも描けるよ。

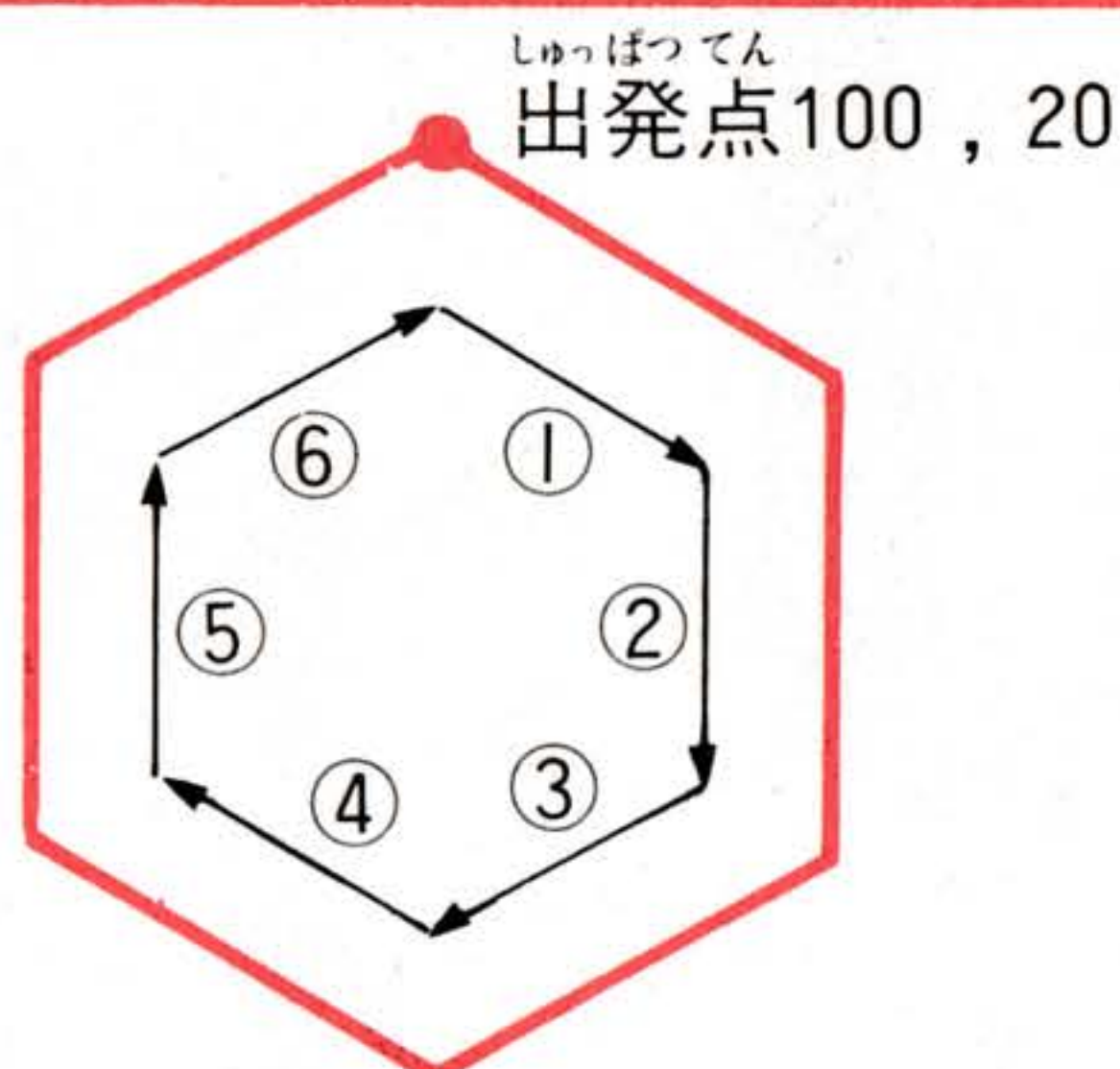
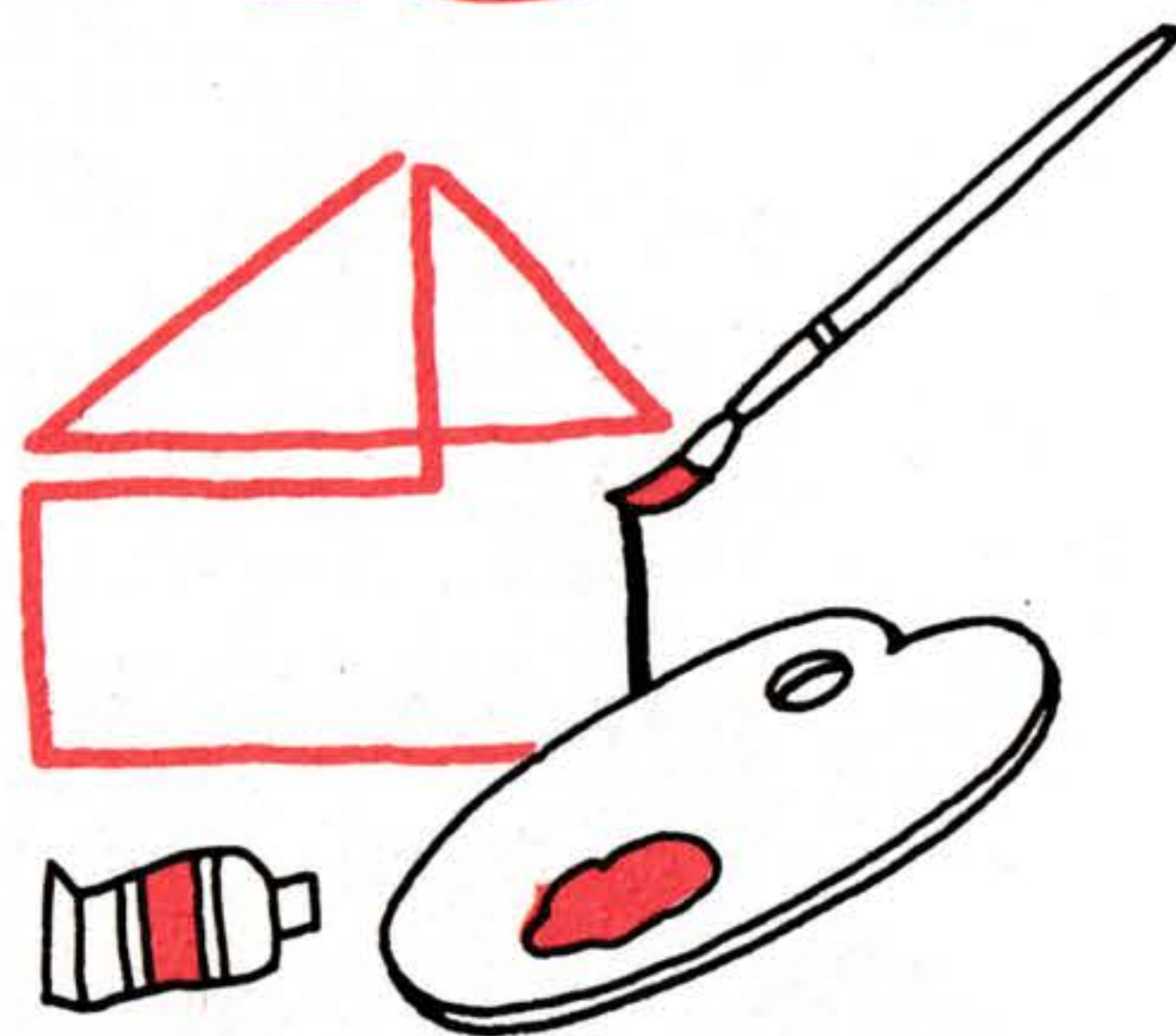
DRAW “文字式”

という文型だから、「文字式」のところで、ペンを動かす方向や長さ、色などを指定してやる。

次のプログラムを入力して実行してみよう。

```
10 SCREEN 2
20 DRAW"C8BM100,20"
30 DRAW"F50D50G50H50U50E50"
40 GOTO40
```

ド ロ ウ めいれい
DRAW命令は
ひと筆描きの要領だ
ひと筆描きでなくても
なんでも描けるけどね



20行のC8はカラーコード8の赤。Bは線を描かずにペンを次の位置まで移動。M100,20は、ペン画を出発させる位置だ。

30行のDRAW文でローマ字は進む方向、数字は進むドット数。

●DRAW文の文字式

まえ
前のページのプログラムを説明してみよう。

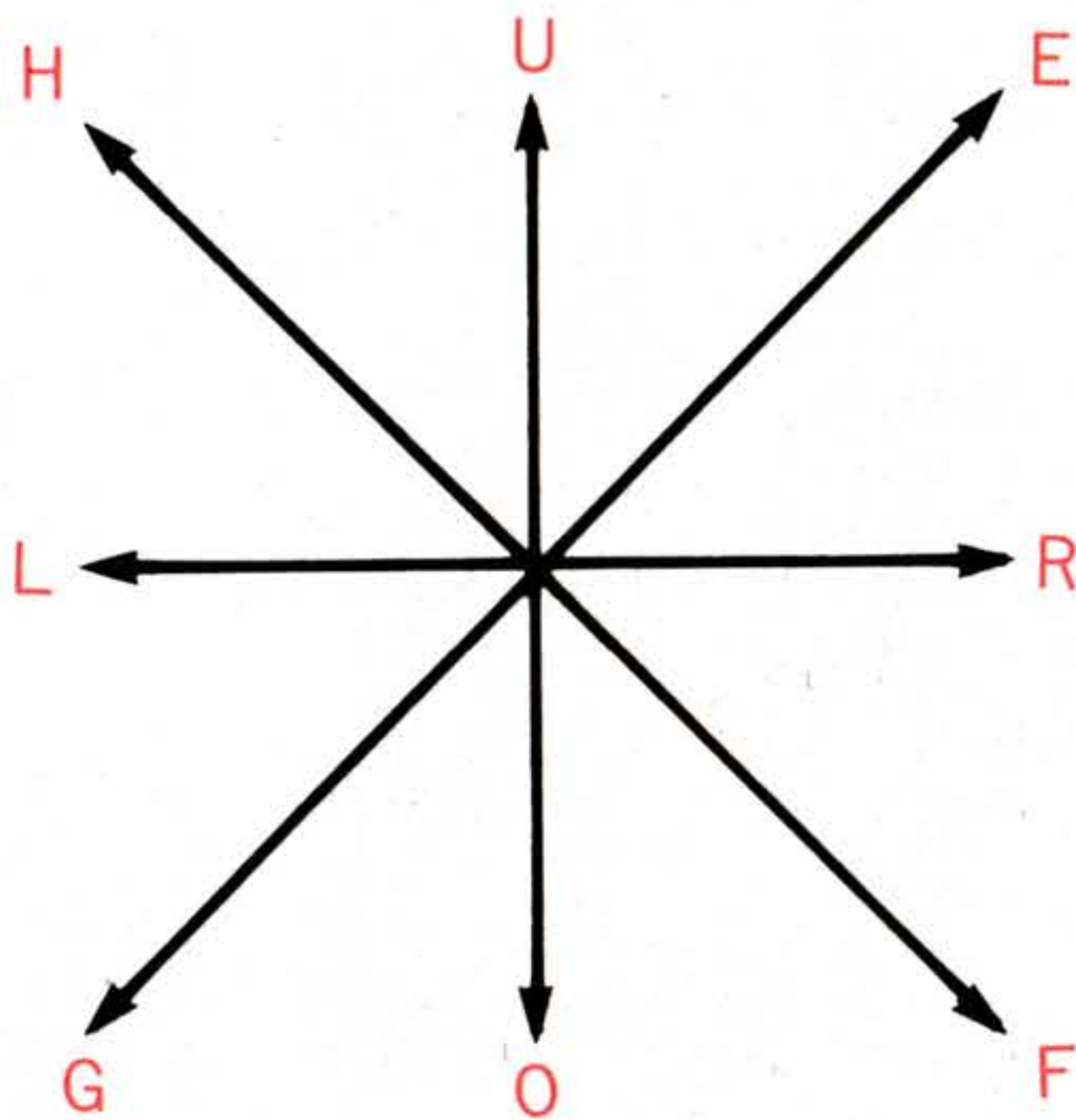
20 DRAW "C 8 BM100, 20"

ペンの出発点の座標。横100、縦20の位置だ
ペンを移動させるが、線は引かない
描く線の色。カラーコードで指定する

30 DRAW "F 50 D 50 G 50 H 50 U 50 E 50"

ローマ字はペンの進む方向。数字は進む長さ

●ペンの移動方向と移動量



ペンは8方向に進められる

U、E、R、F、O、G、L、Hの
8方向だよ

方向の次には引く線の長さを指定だ

U 50なら上の方に50ドット

F 50なら右下の方に50ドット進む

同じドットを指定しても

水平、垂直の線より

斜めの線は実際の距離は長いよ



●DRAW文ショートプログラム

●たくさんのMSXを描く

```
10 SCREEN 2
20 DRAW"BM30,50C8S2A0"
30 GOSUB150
40 DRAW"BM70,50C2S4"
50 GOSUB150
60 DRAW"BM130,50C15S8"
70 GOSUB150
80 DRAW"BM50,130C7S4A1"
90 GOSUB150
100 DRAW"BM150,100C9A2"
110 GOSUB150
120 DRAW"BM180,100C13A3"
130 GOSUB150
140 GOTO 140
150 DRAW"M+4,-16R4M+2,10M+2,-10R4M+3,12"
160 DRAW"R9U2L7U10R18M+3,5M+3,-5R5
170 DRAW"M-5,8M+5,8L5M-3,-5M-3,5L5M+5,-8M-4,-4"
180 DRAW"L10D3R6D9L15M-2,-10M-2,10"
190 DRAW"L4M-2,-10M-2,10L5"
200 FOR I=1 TO1000:NEXT
210 RETURN
```

^{ド ロ ウ} ^{めいれい}
DRAW命令で"MSX"

という文字を、いろいろ変
化させて表示している。

^{ひょうじ} ^{ばしょ} ^{もじ} ^{おお}
表示場所や文字の大きさ、

あるいは表示の向きを変え

たり、いろいろできるとい

うことをわかってくれれば

いいと思うよ。

"MSX"という文字を書かせる^{ド ロ ウ} ^{めいれい}
DRAW命令は、
150行~190行までの^{ド ロ ウ} ^{ぶん}
DRAW文。

ただ、このプログラムは、"MSX"という文字
を表示させるというだけじゃない。どんなふう
に表示させるかということを、20行~120行のとびと
びの^{ド ロ ウ} ^{ぶん} ^{せつめい}
DRAW文で説明している。

座標の指定にマイナス(-)やプラス(+)がつい
ているのは、^{まえ} ^{ざひょう} ^{ほうこう}
前の座標からマイナスやプラス方向
にずれた点という意味だ。



コマンドのSは
スケールファクタ
マクロ命令

Aは
回転マクロ命令だ

4

9

グラフィック画面に 文字を描く



ひら か か ク ロ ー ズ
開いて書き、書いたらCLOSE

オープンめいれい
OPEN命令 ● **ファイルを開く**

グラフィック画面に文字を書くときには、どう
してもしなければならないことがある。次のプロ
グラムを実行してみよう。画面のほぼ中央に赤で
"MSX"、緑色で "BASIC" と表示されただろう。

```
10 SCREEN 2
20 OPEN"GRP:"AS #1
30 COLOR 8
40 LINE(70,50)-(200,120),15,BF
50 PRESET(100,80)
60 PRINT #1,"MSX " ;
70 COLOR 2
80 PRINT #1,"BASIC"
90 CLOSE #1
100 COLOR 15
110 GOTO 110
```

20行のOPEN文をよく見てほしい。

OPEN "GRP:"AS #1

これは、グラフィック画面に文字を表示する
ときは、かならず実行する約束みたいなものだから、
そっくりそのまま暗記してしまおうよ。

開いたものはCLOSEで閉じることになる。

プリント
PRINT #は

ファイルにデータを
書き出す命令だ



グラフィック画面に
白い箱を描いて、
その中に文字を表示
させるプログラムだ



410

パソコン・ミュージック



プレイ めいれい やくそく PLAY命令と約束ごとをおぼえて

プレイ めいれい
PLAY命令 ● " " で囲んだメロディーを演奏する

パソコンに音楽を演奏させる命令が、PLAYだ。

プレイ
PLAYのあとに、" "(ダブルクォーテーション)で
かこ 囲ってメロディーや、演奏に必要な約束ごとを書
くんだ。

ピアノや笛やバイオリン、ぼくたちが楽器を弾
くときは楽譜のオタマジャクシを見て演奏する。

パソコンは楽譜は読めないから、かわりにコン
ピュータ用のミュージック言語という特別なこと
ばを使って、音楽を演奏させるんだ。

ちゃんとした音楽が演奏されるためには、音階、
音の長さ、オクターブ、テンポ、休み、音の大き
さなど、いろいろな要素がある。これらをすべて
パソコンがわかるように、ミュージック言語にな
おしてやらなければならない。

たとえば、音階の「ドレミファソラシ」は、ミ
ュージック言語では「CDEFGAB」、4分休符はR
4というぐあいだ。

まず、この約束ごとをおぼえてしまおう。

MSXは、最高3重
和音まで演奏できる
PLAYのあとに

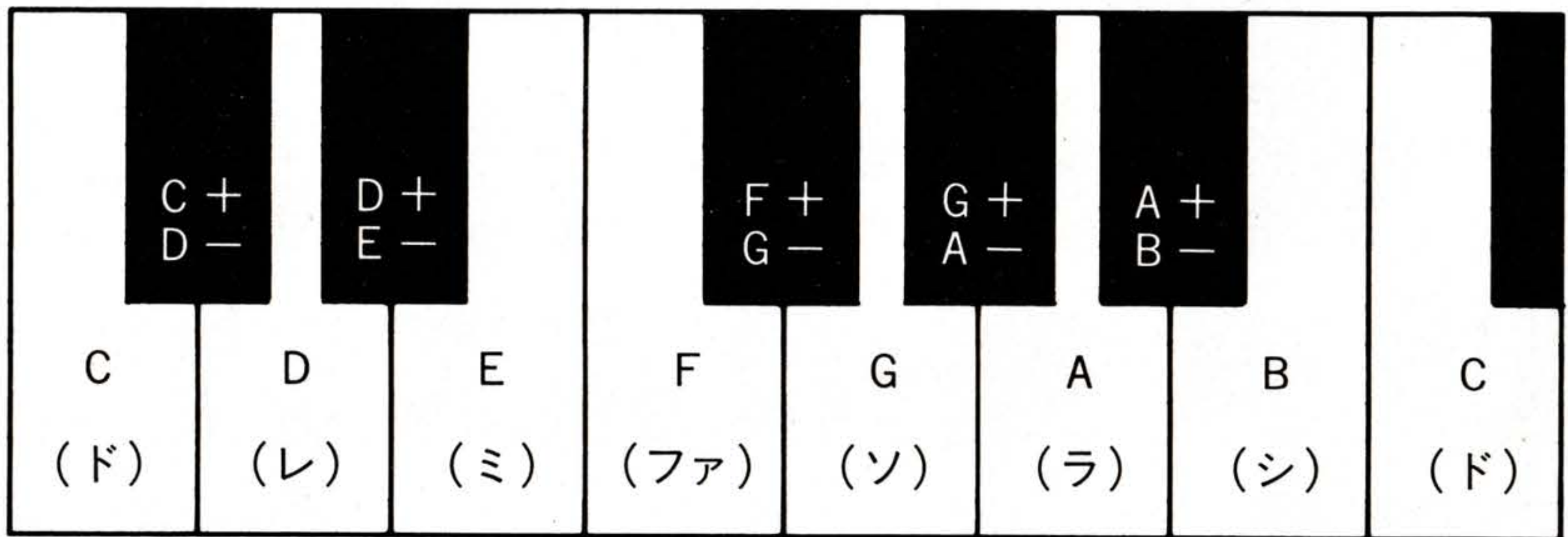
" " を3つ、(コ
ンマ) でつなげばい
いんだ



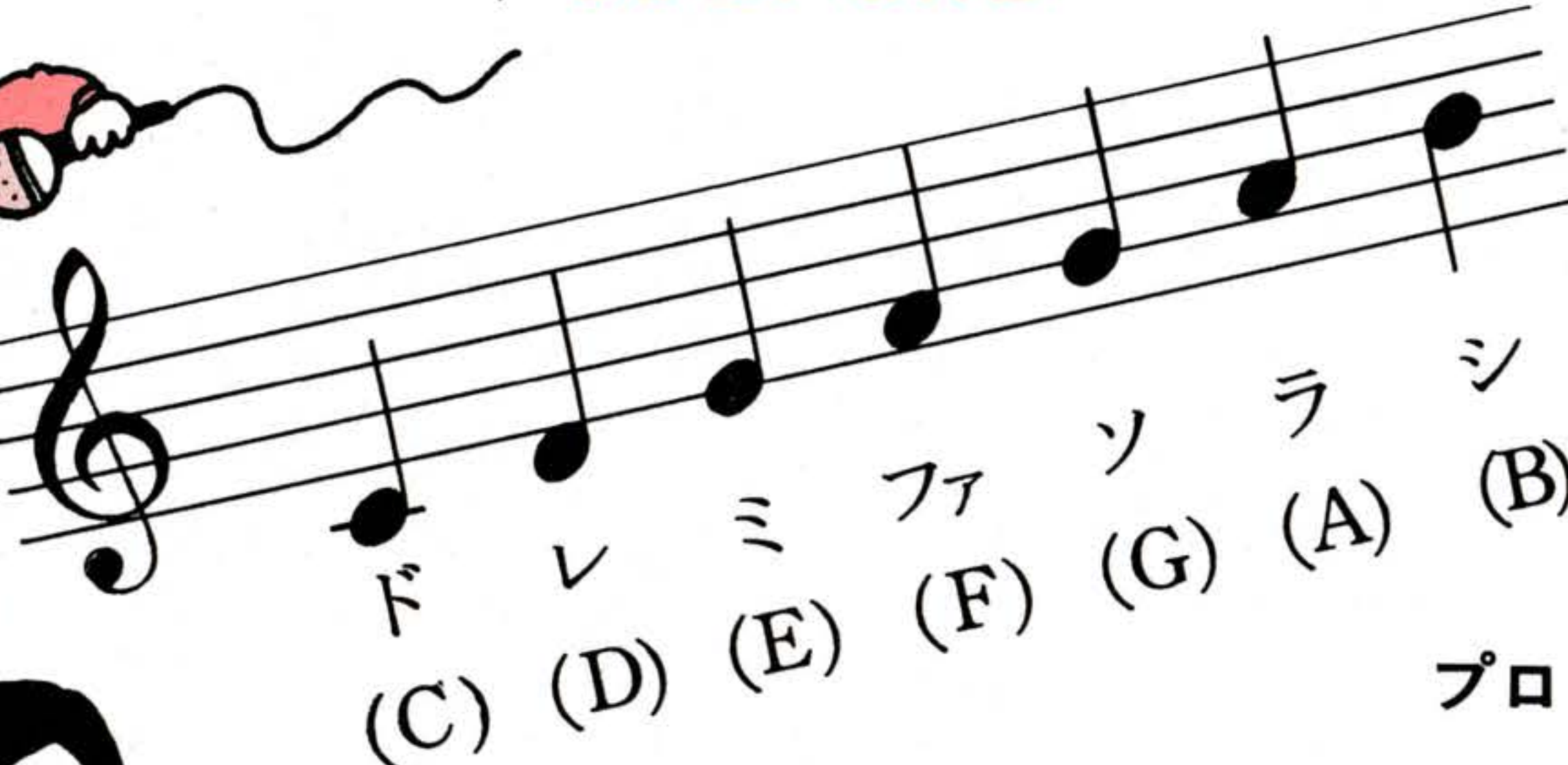
きみが音ちでも心配
ないよ。正しい命令
を与えてやれば、パ
ソコンがちゃんと演
奏してくれるさ



●CDEFGAB



ドレミファソラシは、CDEFGAB



プログラムにすると、

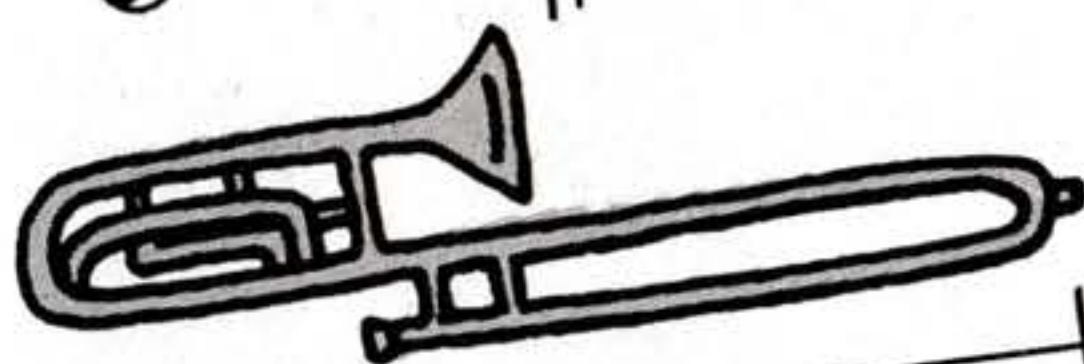
PLAY "CDEFGAB" だ



PLAY "CC+DD+EFF+GG+AA+B"



プログラムにすると、



PLAY "O5CO4BB-AA-GG-FEE-DD-C"



プログラムにすると、






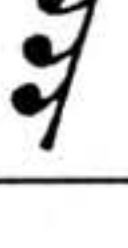
ドレミファソラシの音階はCDEFGAB。半音上げるときは音階のうしろに+ (プラス) または#、下げるときは- (マイナス) 記号をつける。

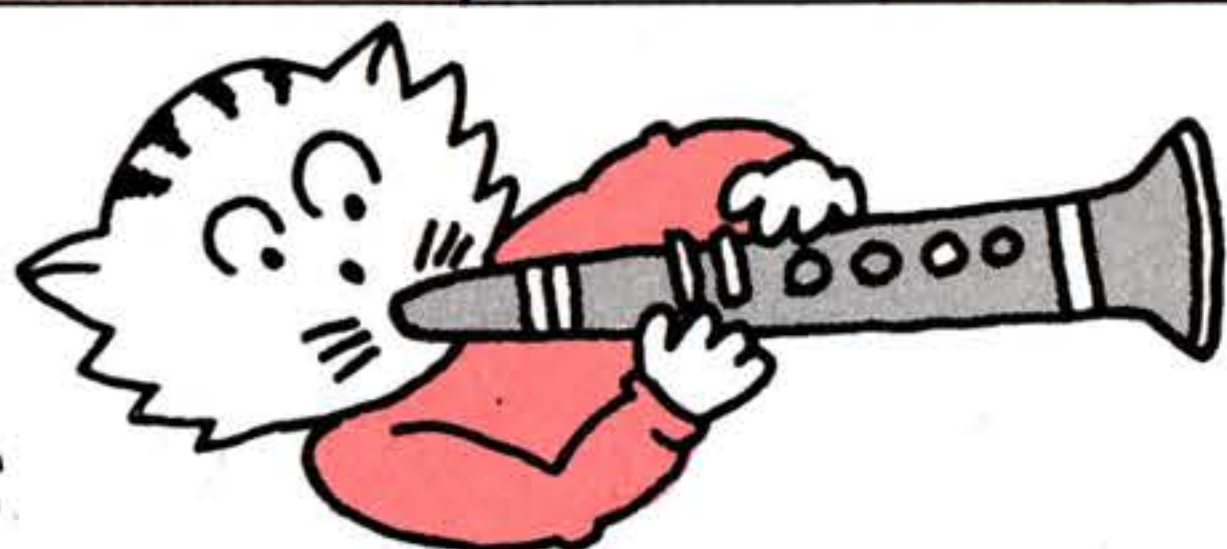
●音の長さや休みの長さ

音の長さ

ぜん おん ぶ 全音符		L 1
ふ おん ぶ 2 分音符		L 2
ふ おん ぶ 4 分音符		L 4
ふ おん ぶ 8 分音符		L 8
ふ おん ぶ 16 分音符		L 16
ふ おん ぶ 32 分音符		L 32
ふ おん ぶ 64 分音符		L 64

休みの長さ

ぜん きゅう ぶ 全休符		R 1
ふ きゅう ぶ 2 分休符		R 2
ふ きゅう ぶ 4 分休符		R 4
ふ きゅう ぶ 8 分休符		R 8
ふ きゅう ぶ 16 分休符		R 16
ふ きゅう ぶ 32 分休符		R 32
ふ きゅう ぶ 64 分休符		R 64

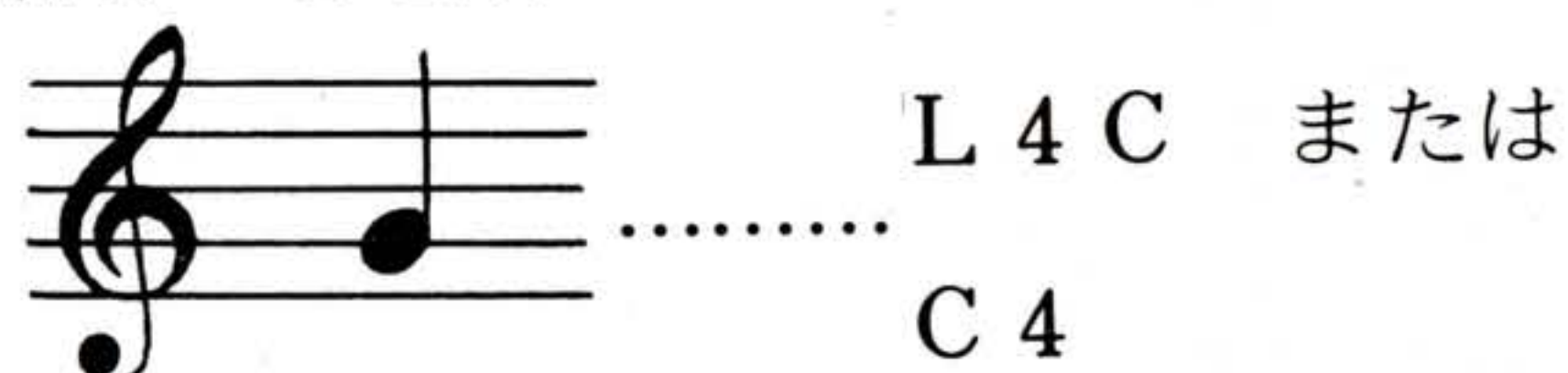


音の長さは、アルファベットのLと数字であらわす。Lは英語のLength（長さ）の頭文字だ。

音の長さはL 1からL 64までである。数が大きくなるほど、短い音になるよ。

書き方は2つあって、どちらもOK。

(例) 4 分音符のソ



符点音符には、. (ピリオド)をつける。

(例) 付点4 分音符のソ



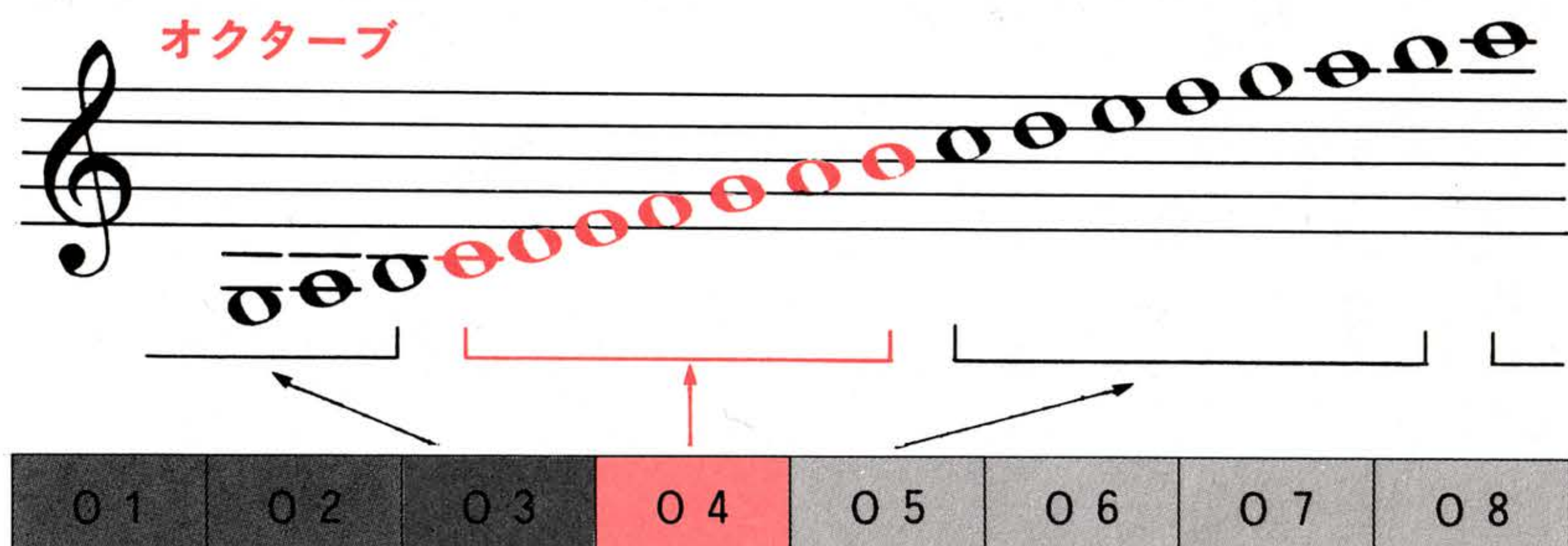
付点2 分音符のソ



Lの記号をRに変えると、休みの長さだ。RはRest（休み）の頭文字で、Lと同じく、数が大きいほど休みの長さは短くなるよ。

パソコンに何も指定しなければ、音の長さはL 4、休みの長さはR 4にセットされている。

●オクターブ、テンポ、ボリューム



低いオクターブ ← 中心のオクターブ → 高いオクターブ

音階はドレミファソラシの7つの音からできているが、同じドでもすごく高いドもあれば、低いドもあるね。

この音の高さを指定するのがオクターブ (Octave) の頭文字、Oだ。

いちばん低いオクターブO1から、いちばん高いオクターブO8までが出せる。

テンポ

曲の速さを指定するのが、テンポ (Tempo) の頭文字、Tだ。

Tは、32から255までの数字で指定し、数字が小さいほどゆっくりしたテンポ、大きくなるほど速いテンポで演奏するんだ。

IO PLAY "O3 CDEFGAB"

IO PLAY "O4 CDEFGAB"

IO PLAY "O5 CDEFGAB"

オクターブを3つ変えてみた。同じようにVやTに変えて、違いをためてみよう。

ボリューム

ボリューム (Volume) は音の大きさだ。Vで指定して、そのうしろに0から15までの数字をつける。数字が大きくなるほど音は大きくなる。

パソコンに何も指定しなければ、それぞれ、O4、T120、V8にセットされている。



ミュージック・プログラミング

♩=180



ロンドンばし おちた おちた おちた



ロンドンばし おちた どー う しょう

ミュージック言語をまとめて説明したので、実際にかんたんな曲を演奏してみよう。イギリス民謡の「ロンドン橋」だ。

譜を見ると、8分音符と16分音符、4分音符、4分休符の組み合わせだ。

まず、テンポ、オクターブ、ボリュームの確認だ。曲の頭にこれを設定しておこう。

10 C L S

20 P L A Y "T 180 O 4 V 8"

と一行にまとめて設定しておこう。

次は、30行に1小節目をプログラミングしよう。

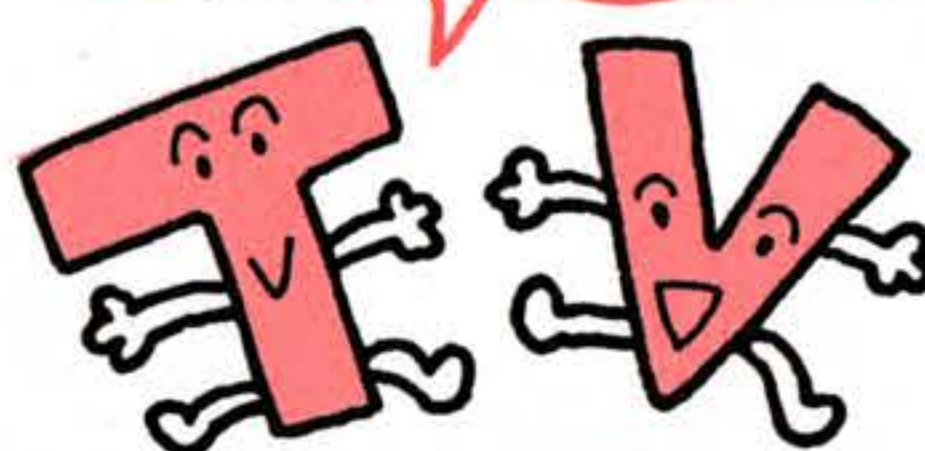


30 P L A Y "L 8 G. L 16 A L 8 G F"



40 P L A Y "E F L 4 G"

テンポやボリュームをいろいろ変えてためてみよう



音の長さは、変わるたびに指定しなす



●音の長さ、オクターブは変わるたびに指定する

のこ 残りを自分でプログラミングしてみよう。

```
10 CLS:LOCATE9,10
20 PRINT"ロト"ン ハ"シ"
30 PLAY"T18004V8"
40 PLAY"L8G.L16AL8"
50 PLAY"GFEFL4G"
60 PLAY"L8DEL4F"
70 PLAY"L8EFL4G"
80 PLAY"L8G.L16AL8GF"
90 PLAY"EFL4G"
100 PLAY"DGL8ECR4"
110 GOTO 30
```

プログラミングの注意 1

音の長さは、変わるたびに指定しなおす。指定を忘れると、前に指定された長さで演奏される。



プログラミングの注意 2

オクターブも、変わるたびに指定しなおす。指定を忘れると、前に指定されたオクターブで演奏される。



PLAY "L4GAG05C04G
AL2GL4AAGA"



PLAY "LI605ED+ED+E
O4BO5DCO4L2A"

休符を利用して同じ音を区切る

パソコンは楽器とちがって、同じ音を切って演奏することができない。同じ音が続くときは、休符を使って区切ってやろう。

プログラミングの注意 3

半音上げる（+または#）、半音下げる（-）指定は、そのたびに指定する。



PLAY "GEE2L4FDD2"

PLAY "GE8R8E2L4FD8
R8D2"





3つの音まで同時に出せる

ひとつの音なら、PLAYのあとに “ ” がひとつ、2つの音を同時に出すなら “ ” , “ ”、3重和音なら、 “ ” , “ ” , “ ” だ。

“ ” と “ ” は、かならず、(コンマ) でつなぐこと。4つつなぐとエラーだよ。



2重和音なら

PLAY “CDE”, “EFG”



3重和音なら

PLAY “CC03B”, “EFG”, “GFG”

次の楽譜は、モーツァルトのトルコ行進曲の出だした。復習のつもりでプログラムにしてみよう。



Allegretto



プログラム100行 110行 120行 130行 140行 150行 160行 170行

コンピュータでは、小節に区切る必要はないから、自分で作りやすいように切ってプログラムにすればいい。RUNさせて、うまく演奏されないときは、前の約束ごとをもう一度読み直そう。



●トルコ行進曲のプログラム

```

10 CLS
20 SCREEN 0,0
30 LOCATE 12,10:PRINT"*****"
40 LOCATE 12,11:PRINT"*"
50 LOCATE 12,12:PRINT"* トルコ マーチ *"
60 LOCATE 12,13:PRINT"*"
70 LOCATE 12,14:PRINT"*****"
80 PLAY "T120V8","T120V8","T120V8"
90 FOR I=0 TO 1
100 PLAY "05L16BAG#A","R4","R4"
110 PLAY "C8R8","03L16AR1604CR16",
      "04L16ER1604ER16"
120 PLAY "05L16DC04B05C","04L16CR16CR16",
      "04L16ER1604ER16"
130 PLAY "05E8R8","03L16AR1604CR16",
      "03L16AR1604ER16"
140 PLAY "05L16FED#E","04L16CR16CR16",
      "04L16ER16ER16"
150 PLAY "05L16BAG#A","03L16AR1604CR16",
      "03L16AR1604ER16"
160 PLAY "05L16BAG#A","03L16AR1604CR16",
      "03L16AR1604ER16"
170 PLAY "06L4C","03L16AR1604CR16",
      "03L16AR1604CR16"
180 PLAY "05L8A06C","04L16CR16CR16",
      "04L16ER16ER16"
190 PLAY "05L8BAGA","05L8BF#EF#",
      "03L16ER16AR16AR16AR16"
200 PLAY "05L8BAGA","05L8BF#EF#",
      "03L16ER16AR16AR16AR16"
210 PLAY "05L8BAGF#E4","05L8BF#ED#E4",
      "03L16ER16AR1602AR1603AR16E4"
220 NEXT I
230 FOR I=0 TO 1
240 PLAY "05L8EF","05L8CD","R4"
250 PLAY "05L16GR16GR16AGFE","05L16ER16ER16R4",
      "03L8C04C03E04E"
260 PLAY "05L4DE8F8","04L8AG05CD","03GR4"
270 PLAY "05L16GR16GR16AGFE","05L16ER16ER16R4",
      "03L8C04C03E04E"

```



```

280 PLAY "03L4DC8D8", "04L4BA8B8", "03G4R4"
290 PLAY "05L16ER16ER16FEDC", "05L16CR16CR16R4",
      "02L8A03AC04C"
300 PLAY "04L4B03C8D8", "04L8G#EAB", "03L4ER4"
310 PLAY "05L16ER16ER16FEDC", "05L16CR16CR16R4",
      "02L8A03AC04C"
320 PLAY "04L4B", "04L4G#", "03E"
330 PLAY "L16BAG#A", "R4", "R4"
340 PLAY "05C8R8", "03L16R1604CR16",
      "03L16AR1604ER16"
350 PLAY "05L16DC04B05C", "04L16CR16CR16",
      "04L16ER16ER16"
360 PLAY "05E8R8", "03L16AR1604CR16",
      "03L16AR1604ER16"
370 PLAY "05L16FED#E", "04L16CR16CR16",
      "04L16ER16ER16"
380 PLAY "05L16BAG#A", "03L16AR1604CR16",
      "03L16AR1604ER16"
390 PLAY "05L16BAG#A", "03L16AR1604CR16",
      "03L16AR1604ER16"
400 PLAY "06C405A8B8", "03L16FR16AR16AR16AR16",
      "03L16F04D#R16D#R16D#R16"
410 PLAY "06L8C05BAG#", "03L16ER1604ER1603DR16BR16",
      "03L16ER16AR16DR16FR16"
420 PLAY "05L8AEFD", "03L16CR16AR16DR16BR16",
      "03L16CR16ER16DR16FR16"
430 PLAY "05C404L8C.A32B32A4",
      "03L16AR16AR16G#R16G#R16A4",
      "03L16ER16ER16ER16ER1602A4"
440 NEXT I

```



4

11

楽しいサウンド作り



すう ち か
数値を変えてためしてみよう

SOUND命令 ● サウンドを発生させる

サウンドを作るのが、SOUND命令だ。この命令を使うと、PLAY命令で作った音とはちがった、もっと細かい音まで出せるんだ。

ただ、この機能の説明は、初心者の人にとってはかなり複雑でむずかしいのだ。

そこで、サウンドの書き方をきみに教えるから、あとは自分で数値を変えながらためしてみるのだ。気に入った音が出たら、メモにとっておいて、ゲームに使うのもいい。

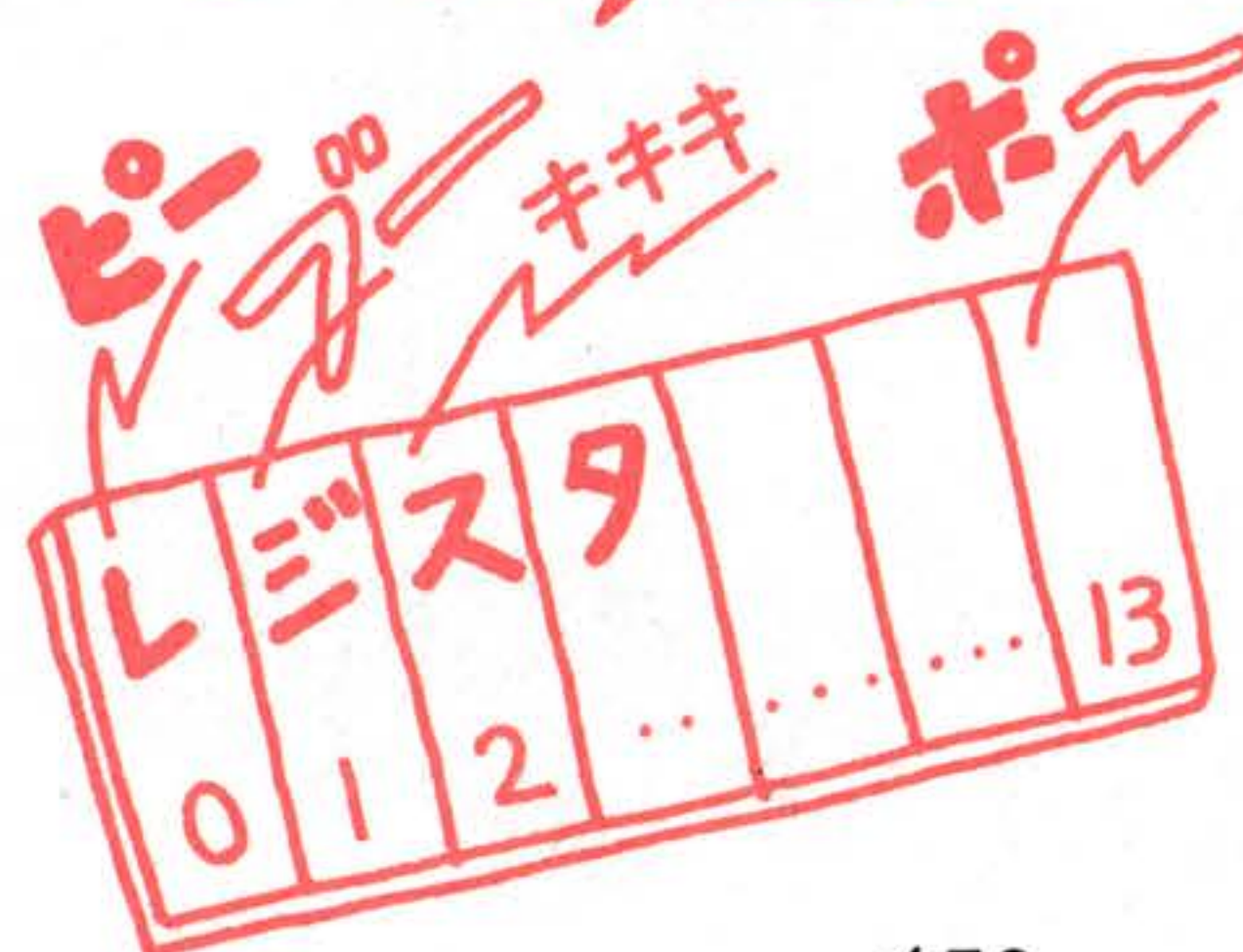
SOUND レジスタ番号, データ

レジスタというのは、音を発生させるところで、0から13まで、ぜんぶで14の発生源がある。同じレジスタでも、与えるデータによって鳴る音が違うんだ。データには数値を入れるのだけど、0から13までのそれぞれのレジスタに入れられる数値の範囲は決まっている。次のページに、その範囲を書いておいたから、この範囲を超えないように注意しよう。

ゲームをおもしろくするのも、つまらなくするのもサウンドしだい。シーンとしたゲームなんて、スリルも迫力もない

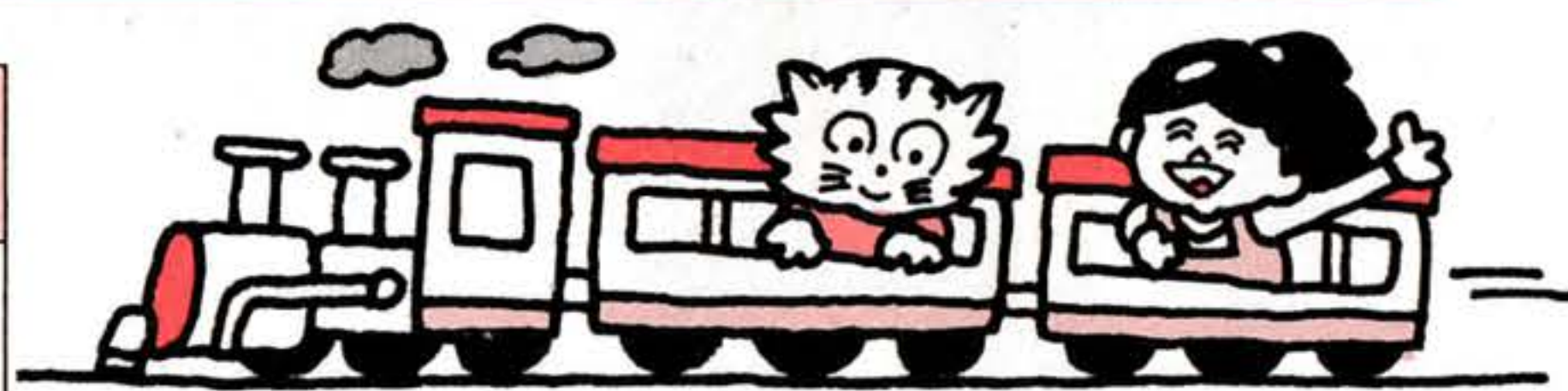


各レジスタには、それぞれいろいろな役わりがあるんだよ



●発生源のレジスタ番号とデータを指定して組み合わせる

レジスタ番号	データの範囲
0	0 ~ 255
1	0 ~ 15
2	0 ~ 255
3	0 ~ 15
4	0 ~ 255
5	0 ~ 15
6	0 ~ 31
7	0 ~ 63
8	0 ~ 16
9	0 ~ 16
10	0 ~ 16
11	0 ~ 255
12	0 ~ 255
13	0 ~ 15



サウンド「^{きしゃ}汽車」

10	SOUND	6, 13
20	SOUND	7, 35
30	SOUND	8, 16
40	SOUND	9, 16
50	SOUND	11, 100
60	SOUND	12, 1
70	SOUND	13, 10

^{ひだり}左がレジスタの^{ばんごう}番号、^{みぎ}右がその
^{すうち}数値だ。^{おな}同じ^{きしゃ}汽車でも、^{つか}使うレジ
スタやデータの^{すうち}数値を^か変えようと、
^{さか}坂を^{のぼ}登っている^{おと}音がでるよ。

レジスタは14こあるけど、ぜんぶ
のレジスタを^{つか}使わなければいけない
ということはないよ。3つや4つの
レジスタを^{つか}使って^{つく}作る^{おと}音もあるんだ。
^{きしゃ}汽車は7つのレジスタでできる。



サウンド「^{のぼ}登り^{さか}坂の^{きしゃ}汽車」

10	SOUND	0, 200
20	SOUND	7, 35
30	SOUND	8, 16
40	SOUND	9, 16
50	SOUND	11, 100
60	SOUND	12, 4
70	SOUND	13, 14

●サウンド作りプログラム

最後に、数値を入力するだけで、
いろいろなサウンドが飛び出すプロ
グラムをのせておこう。

プログラムの80行から 210 行まで
に各レジスタ番号とその役わりが書
いてあるね。ここには、音の鳴らし
方や音の大きさなどいろいろな機能
がある。とにかくどんどん数値を入
れてみよう。

10	SOUND	1, 3
20	SOUND	8, 16
30	SOUND	12, 10
40	SOUND	13, 8

ドラムの音のプ
ログラムはこう
だ。ためしに入
れてごらん



```

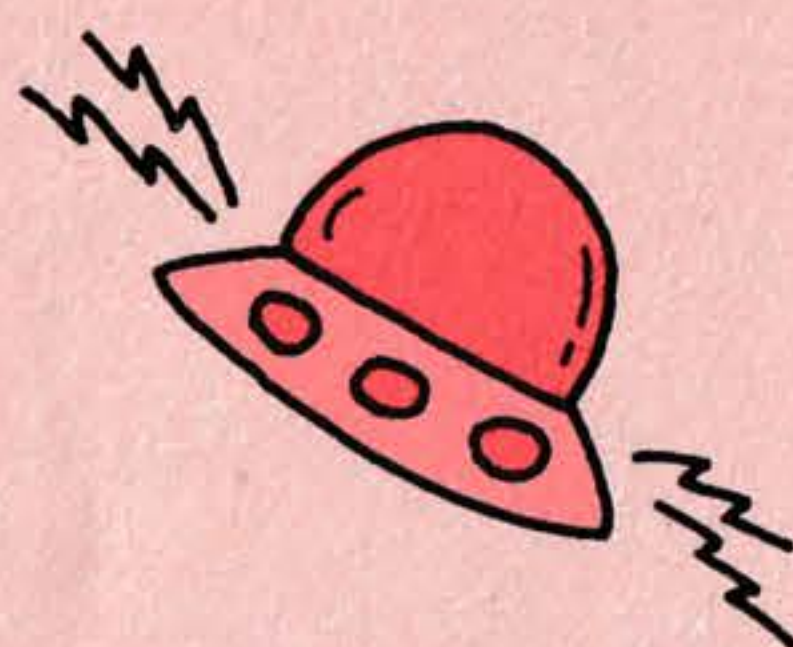
10 CLS
20 KEY OFF
30 DIM A(13)
40 PRINT"*****"
50 PRINT" * SOUND EDITER *"
60 PRINT"*****"
70 PRINT"SOUND No."
80 PRINT" 0.....Aチャンネル(0-255)  ::"
90 PRINT" 1.....Aチャンネル(0-15)  ::"
100 PRINT" 2.....Bチャンネル(0-255)  ::"
110 PRINT" 3.....Bチャンネル(0-15)  ::"
120 PRINT" 4.....Cチャンネル(0-255)  ::"
130 PRINT" 5.....Cチャンネル(0-15)  ::"
140 PRINT" 6.....ノイズ(0-31)  ::"
150 PRINT" 7.....オンゲン(0-63)  ::"
160 PRINT" 8.....Aオンリョウ(0-16)  ::"
170 PRINT" 9.....Bオンリョウ(0-16)  ::"
180 PRINT"10.....Cオンリョウ(0-16)  ::"
190 PRINT"11.....インローフ(0-255)  ::"
200 PRINT"12.....インローフ(0-255)  ::"
210 PRINT"13.....エーバターフ(0-15)  ::"
220 PRINT"-----"
    
```




```

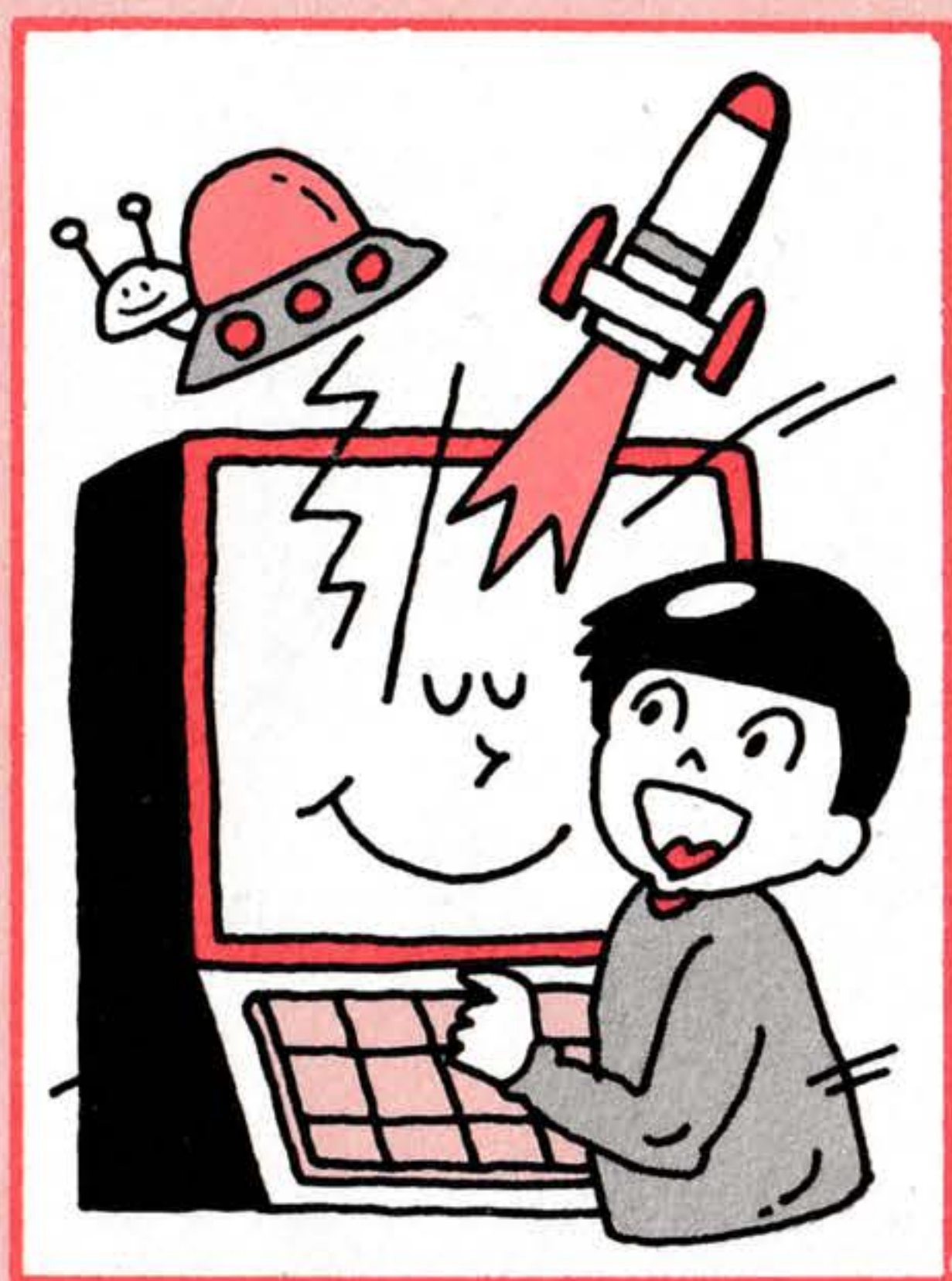
230 FOR N=0 TO 13
240 LOCATE 22, 4+N: INPUT B
250 F=1: GOSUB 450
260 A(N)=B
270 NEXT
280 FOR N=0 TO 13: SOUND N, A(N): NEXT
290 LOCATE 5, 20
300 PRINT "ショウセイ(No, スウチ)"
310 LOCATE 18, 20: INPUT C, D
320 N=C: B=D: F=2: GOSUB 450
330 A(C)=D
340 LOCATE 23, 4+C: PRINT D
350 GOSUB 370
360 GOTO 290
370 FOR N=0 TO 13: SOUND N, A(N): NEXT: RETURN
380 FOR N=0 TO 13: SOUND N, A(N): NEXT: RETURN
390 FORM=0 TO 6: LOCATE 18, 20
400 PRINT "DATA ERROR"
410 FOR I=0 TO 50: NEXT: BEEP
420 LOCATE 18, 20
430 PRINT " "
440 NEXT: RETURN
450 IF N=0 THEN IF B>255 THEN GOSUB 390: H=1
460 IF N=1 THEN IF B>15 THEN GOSUB 390: H=1
470 IF N=2 THEN IF B>255 THEN GOSUB 390: H=1
480 IF N=3 THEN IF B>15 THEN GOSUB 390: H=1
490 IF N=4 THEN IF B>255 THEN GOSUB 390: H=1
500 IF N=5 THEN IF B>15 THEN GOSUB 390: H=1
510 IF N=6 THEN IF B>31 THEN GOSUB 390: H=1
520 IF N=7 THEN IF B>63 THEN GOSUB 390: H=1
530 IF N=8 THEN IF B>16 THEN GOSUB 390: H=1
540 IF N=9 THEN IF B>16 THEN GOSUB 390: H=1
550 IF N=10 THEN IF B>16 THEN GOSUB 390: H=1
560 IF N=11 THEN IF B>255 THEN GOSUB 390: H=1
570 IF N=12 THEN IF B>255 THEN GOSUB 390: H=1
580 IF N=13 THEN IF B>15 THEN GOSUB 390: H=1
590 IF F=1 THEN IF H=1 THEN F=0: H=0: RETURN 240
600 IF F=2 THEN IF H=1 THEN F=0: H=0: RETURN 310
610 F=0: RETURN

```



パート5

ゲームで遊ぼう





8×8の キャラクタを動かす



ロケットを画面に表示させる

スプライトパターン●ゲームのコマ作り

パソコンゲームやテレビゲームには、UFOやインベーダー、ロケットやキャノン砲などのキャラクタがたくさん飛び出してくる。

これらのキャラクタは、いったいどうやって作られているのかなあ、そして、どうすれば、あんなふうに動かすことができるのだろう。

このキャラクタをスプライトパターンというんだ。キャラクタを動かす命令がPUT SPRITE命令というんだ。

まず、キャラクタの作り方から説明しよう。キャラクタは8×8と16×16の2つのタイプがある。16×16の方が、より細かな図柄を表現できるんだ。

はじめに、8×8のかんたんなキャラクタを作ってみよう。

紙に、縦8個横8個の正方形のマスを書く。マス目の左側に、縦に1から8までの番号をふっておく。そして、このマス目を好きなように黒く塗りつぶして、何か形のある図形を作ってごらん。

パソコンのスプライト機能は、作ったキャラクタを画面上に表示して、自由自在に動かすことができる、とても便利な機能なんだ。パソコンゲームには、絶対に欠かせないものだよ



●8×8のSpriteパターンを定義する

1							 & B 0 0 0 0 1 0 0 0
2							 & B 0 0 0 0 1 0 0 0
3							 & B 0 0 0 1 1 1 0 0
4							 & B 0 0 0 1 1 1 0 0
5							 & B 0 1 0 1 1 1 0 1
6							 & B 0 1 1 1 1 1 1 1
7							 & B 0 1 1 1 1 1 1 1
8							 & B 0 1 0 1 1 1 0 1



さて、どんなふうに塗りつぶしたのかをコンピュータに教えてやらなければ、画面には表示できない。

これを「Spriteパターンを定義する」という。

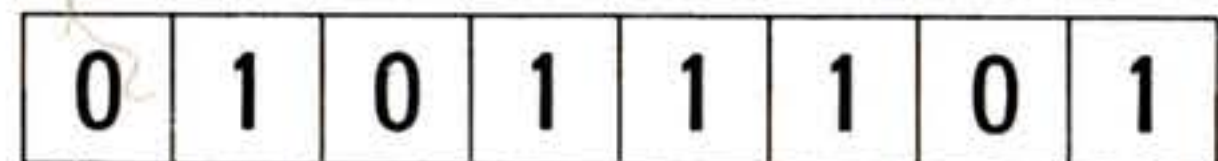
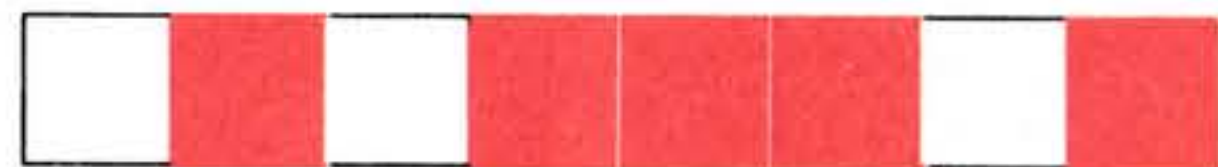
この伝え方は、2進数、10進数、16進数の3通りある。どれを使って、同じ形が表示できるよ。

ここでは、いちばんわかりやすい2進数で説明しよう。

2進数であらわすときは、マス目の白い部分を0、黒く塗りつぶしたところを1にして、Spriteパターンを定義するんだ。

そして、その前に「2進数であらわしているんだよ」というサインの&Bをつける。

たとえば、5段目は

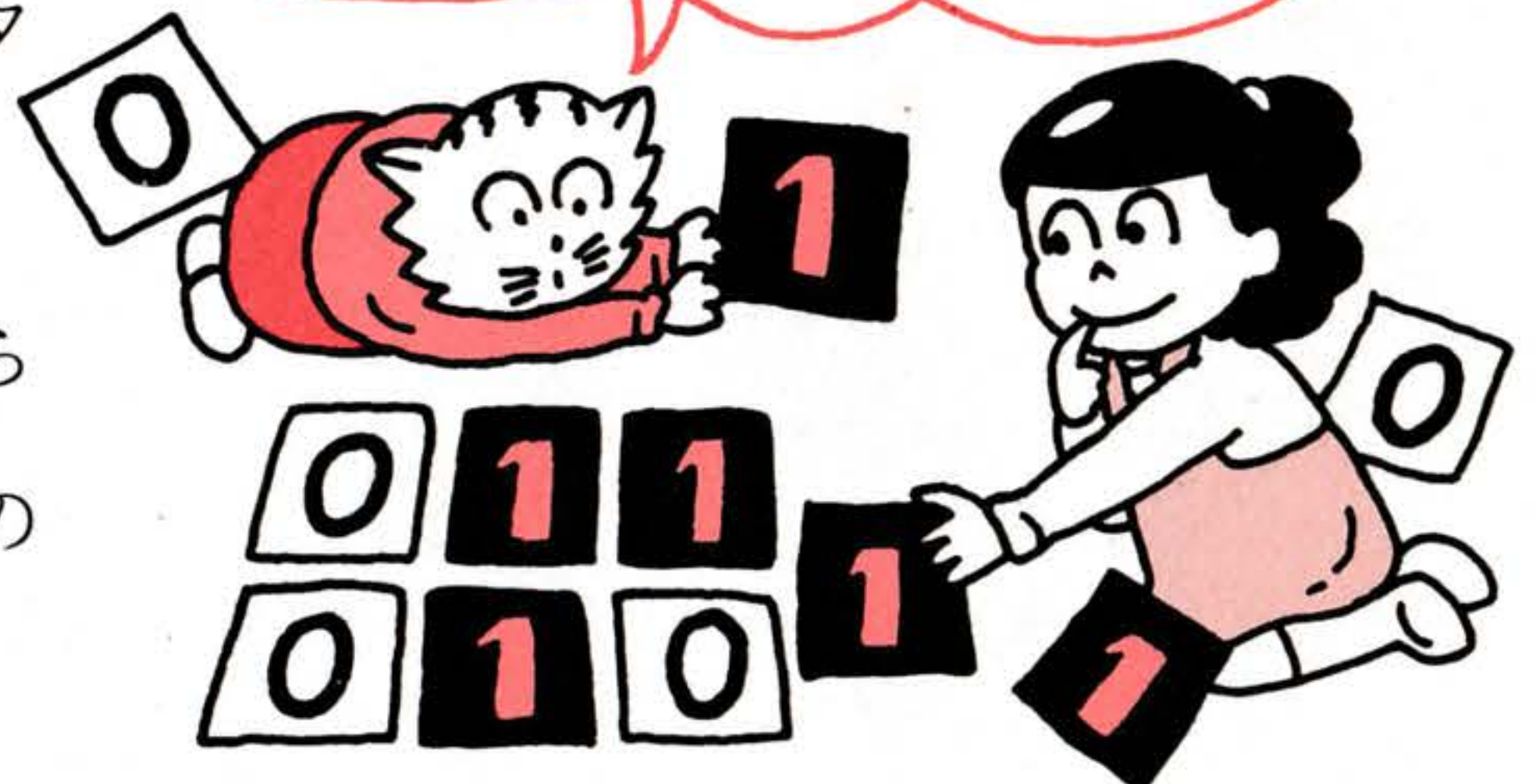


& B 0 1 0 1 1 1 0 1

こうしてマス目の黒と白を数値に置きかえることを、「キャラクターコードに直す」というんだ。書き方は、

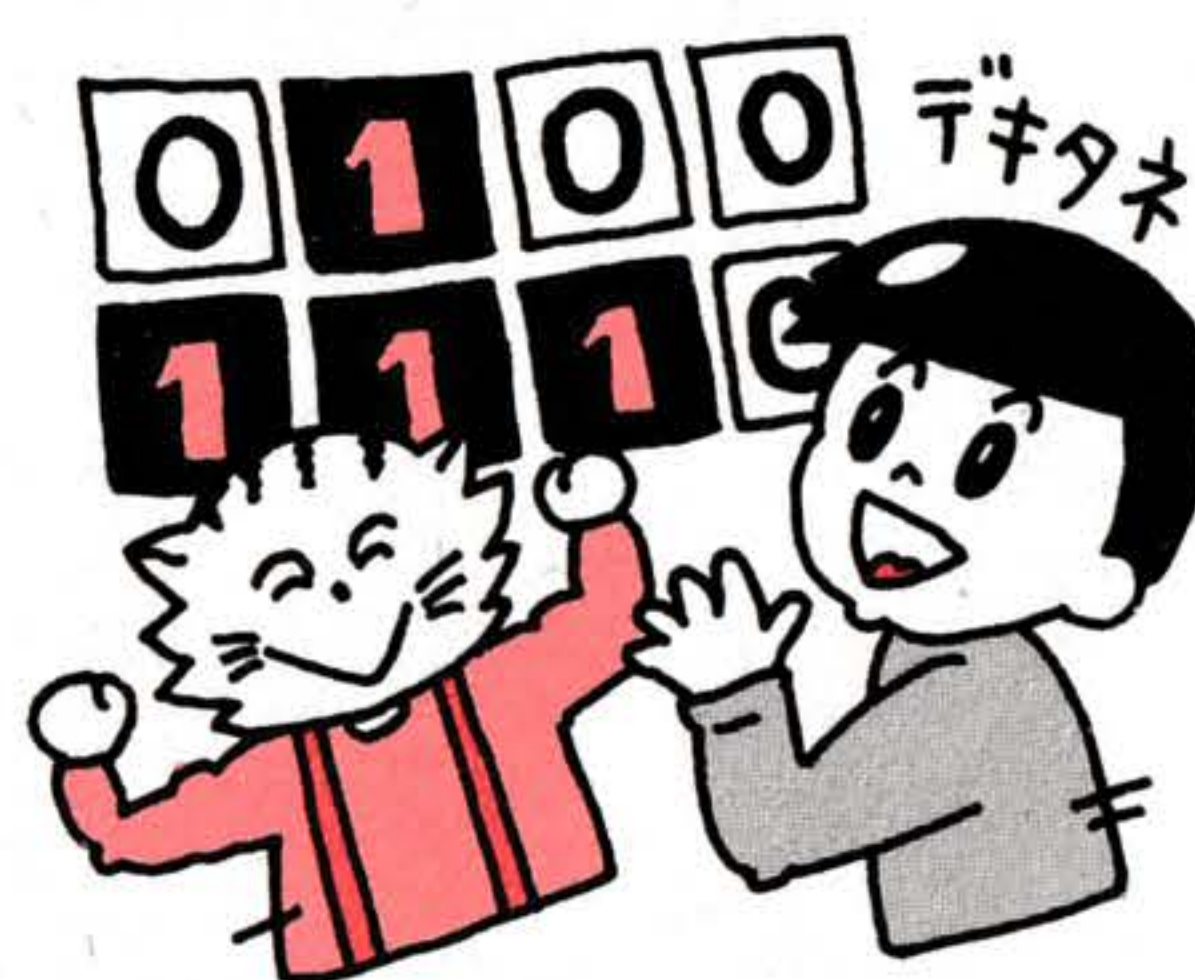
CHR\$ (& B01011101)

となる



S 1 \$										CHR\$ (&B000001000)
S 2 \$										CHR\$ (&B000001000)
S 3 \$										CHR\$ (&B000011100)
S 4 \$										CHR\$ (&B000011100)
S 5 \$										CHR\$ (&B01011101)
S 6 \$										CHR\$ (&B01111111)
S 7 \$										CHR\$ (&B01111111)
S 8 \$										CHR\$ (&B01011101)

マス目の左側の数の意味がわかったかな？
 パターン全体を定義するのを、SPRITE\$と
 いうんだけど、S 1 \$、S 2 \$……は、そ
 のスプライトパターンの1段目、2段目とい
 う意味だ。



パターン全体を定義するには、そう、S 1
 \$からS 8 \$までをたせばいい。

SPRITE\$ (0) = S 1 \$ + S 2 \$ +
 S 3 \$ + S 4 \$ + S 5 \$ + S 6 \$ + S 7 \$
 + S 8 \$

さあ、プログラムに直してみよう。

```

10 SCREEN 1,0
20 S1$=CHR$(&B000001000)
30 S2$=CHR$(&B000001000)
40 S3$=CHR$(&B000011100)
50 S4$=CHR$(&B000011100)
60 S5$=CHR$(&B01011101)
70 S6$=CHR$(&B01111111)
80 S7$=CHR$(&B01111111)
90 S8$=CHR$(&B01011101)
100 SPRITE$(0)=S1$+S2$+S3$+S4$+S5$+S6$+S7$+S8$

```



●スプライトパターンを表示する

スプライトパターンを画面に表示させるには、いくつかのきまりがあるから、おぼえておこう。

●スプライトを表示するときの SCREEN

SCREEN 0は使えない。あとは、テキスト画面でもグラフィック画面でもOK。だから、SCREEN 1 または 2 または 3 だ。



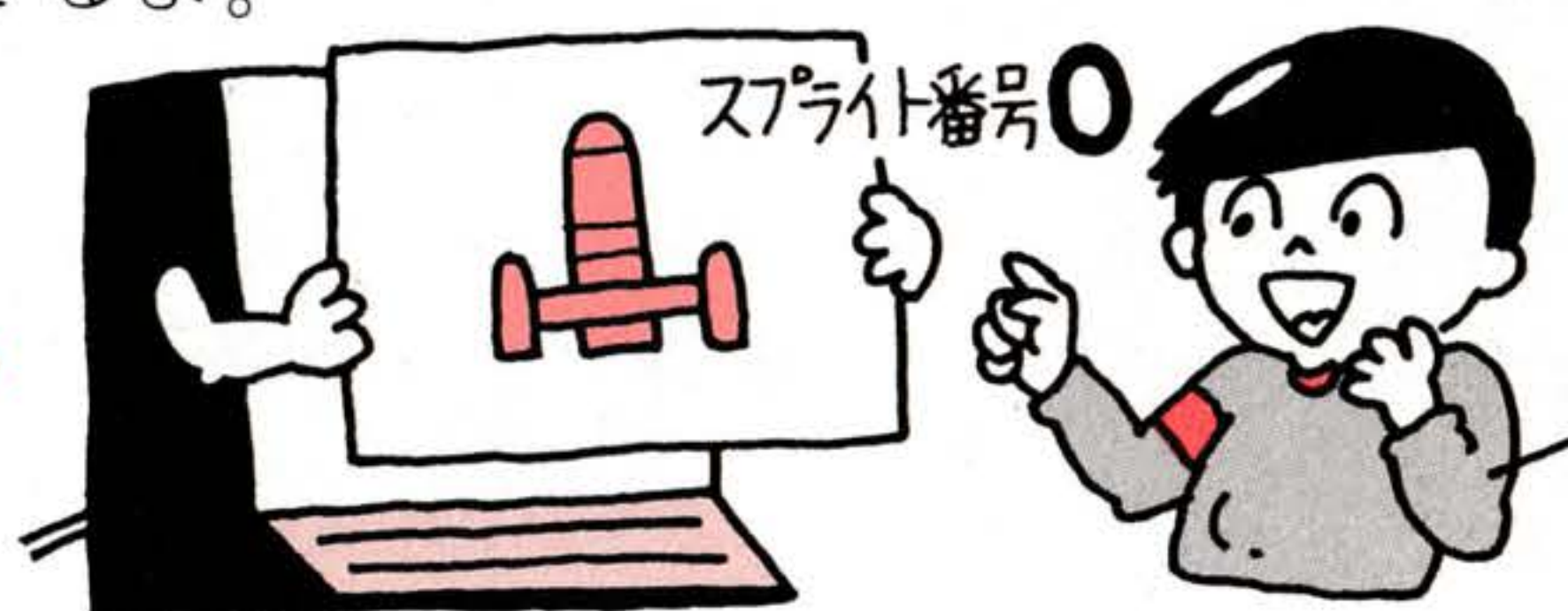
●スプライトのパターン番号

スプライトパターンをいくつも使うときは、どのパターンをどこに定義したのか、番号をつけて区別してやらなければならない。これがスプライト番号で、左ページ

SPRITE\$ (0) =

の0がその番号だ。スプライト番号は、0~255の範囲で設定し、画面に最高256個のパターンが用意できるよ。

このロケットのパターンはスプライト番号0に定義されたんだね



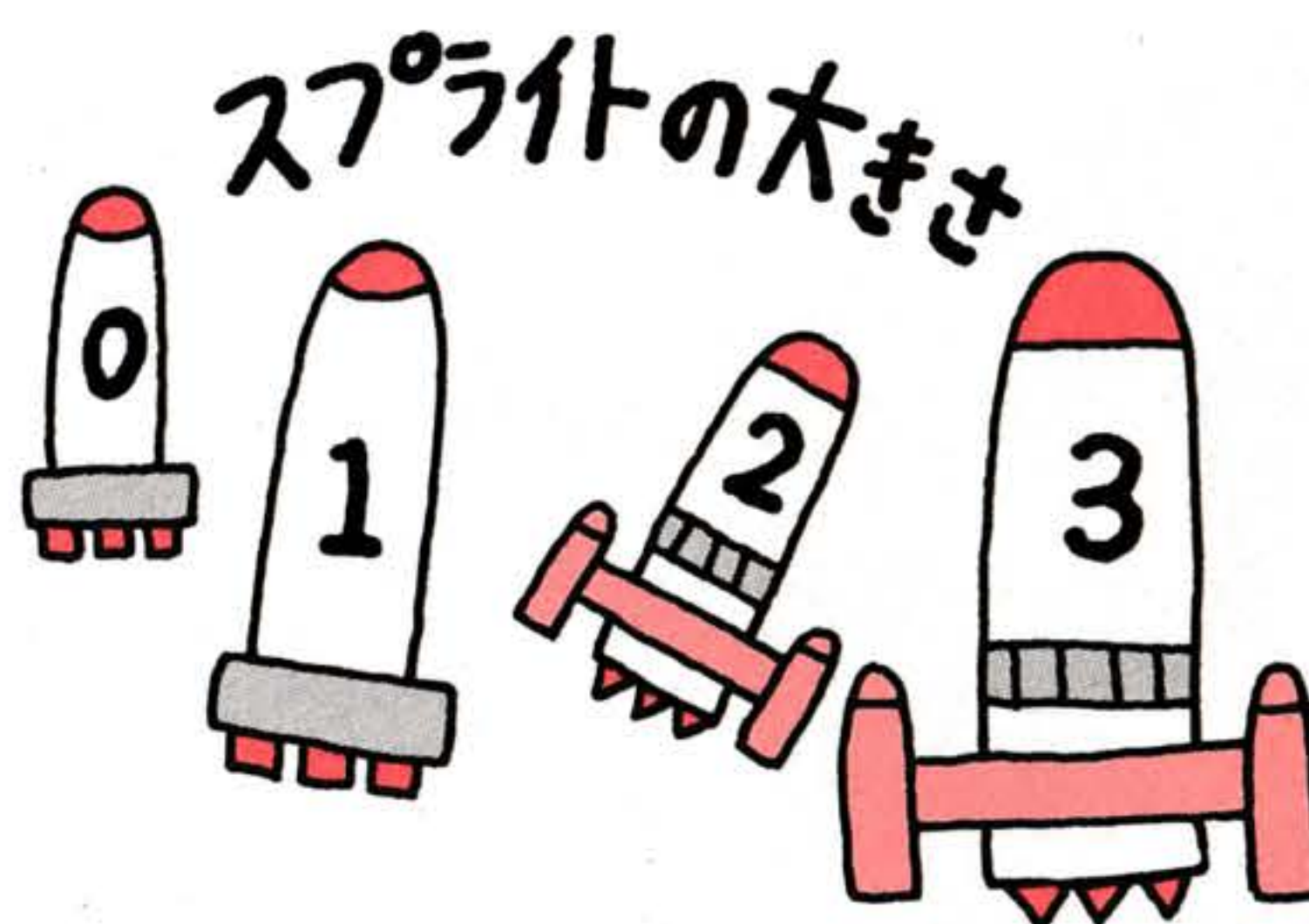
●スプライトの大きさ

0 8×8ドットで文字の大きさと同じ

1 8×8ドットで、縦横の長さが、ふつうの文字の2倍

2 16×16ドット

3 16×16ドットで、縦横の長さがそれぞれ2倍



PUT SPRITE命令 ● スプライトパターンを動かす

スプライトパターンを画面に表示させる命令が、PUT SPRITE命令だ。命令文の書き方は、こうだ。

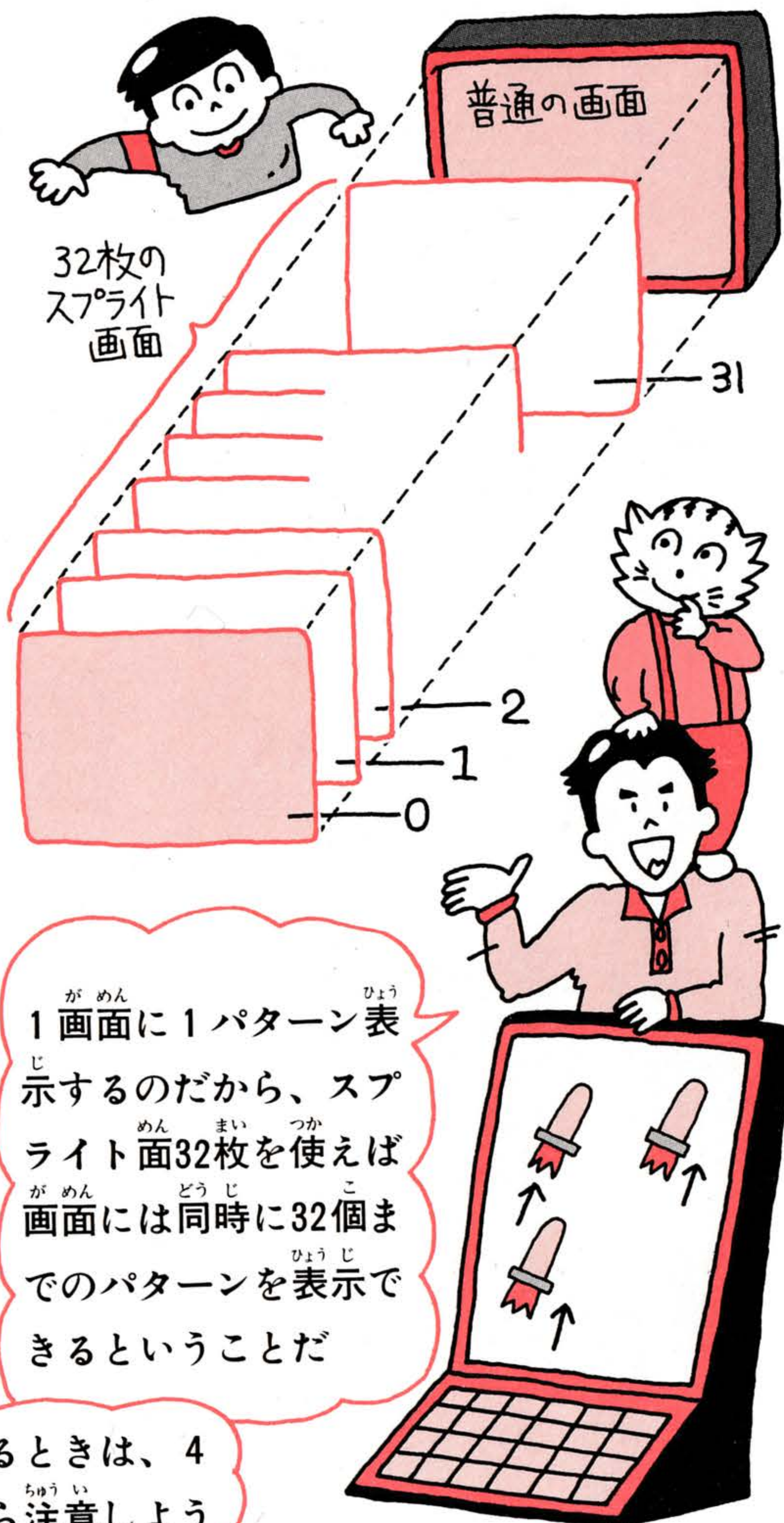
PUT SPRITE スプライト面番号, (表示する位置), 表示する色, スプライト番号

● スプライト面番号

ふつうの画面とは別に、スプライトパターンだけを表示するための専用画面で、0から31まで、ぜんぶで32枚用意されている。

1つのスプライト画面には、1つのスプライトしか表示できない。

同じスプライト画面の別の位置にパターンを表示すると、その瞬間、前のパターンは姿を消してしまう。これを利用すれば、パターンが一瞬のうちに移動したり、変身したかのように表現できるぞ。



1画面に1パターン表示するのだから、スプライト面32枚を使えば画面には同時に32個までのパターンを表示できるということだ

ただし、横に並べて表示するときは、4個までと決められているから注意しよう

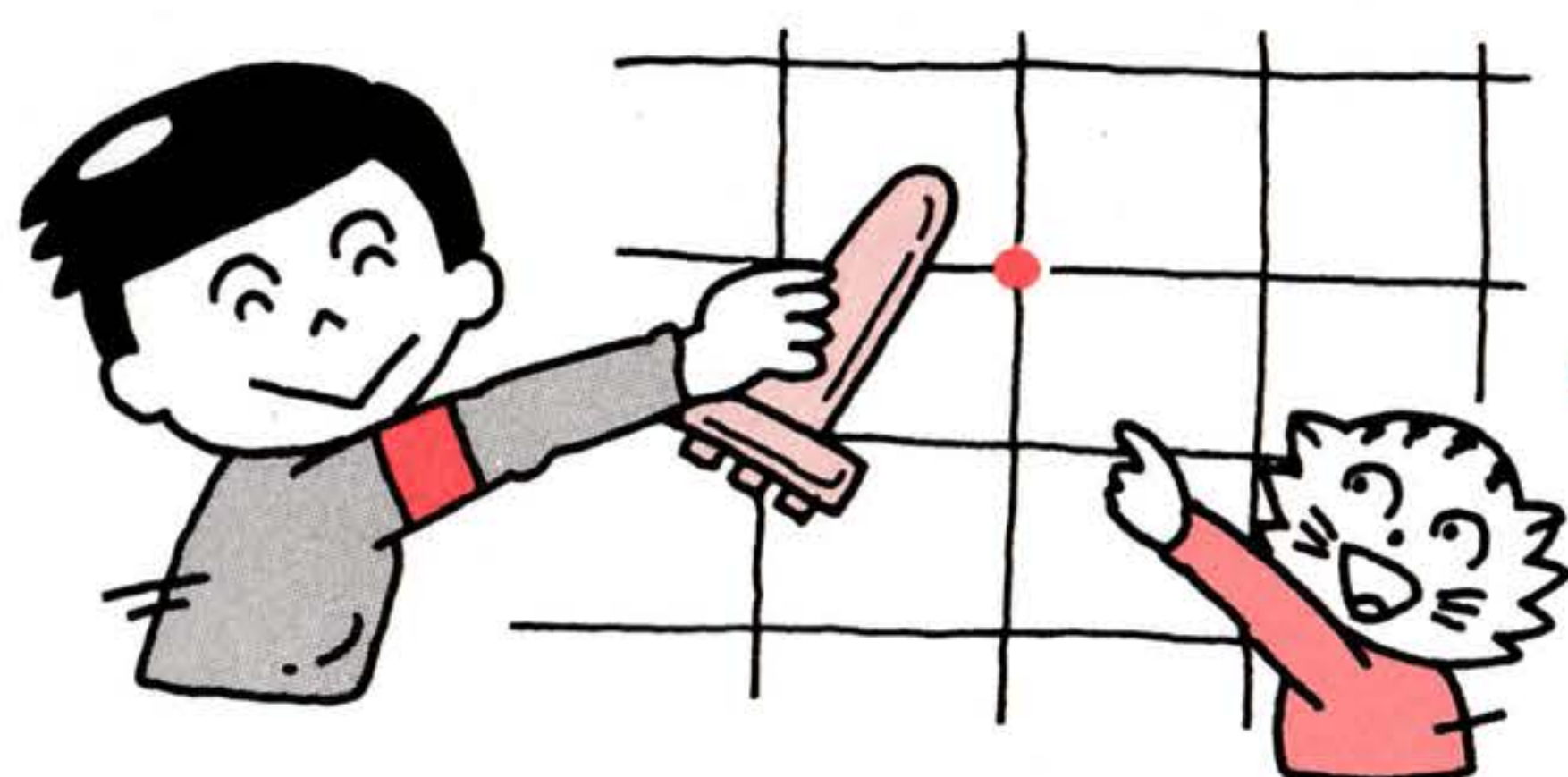
●表示する位置

画面に表示する位置は、
(X座標, Y座標)のこと
だ。X (横の座標) は、
32~255の範囲で、Y (縦の
座標) は-32~191の範囲
で指定するんだ。

さあ、画面に表示させてみよう。
次のプログラムを追加して、R
ENUMでそろえてから、実行し
てごらん。まず、使うSCREEN
と大きさを指定してから



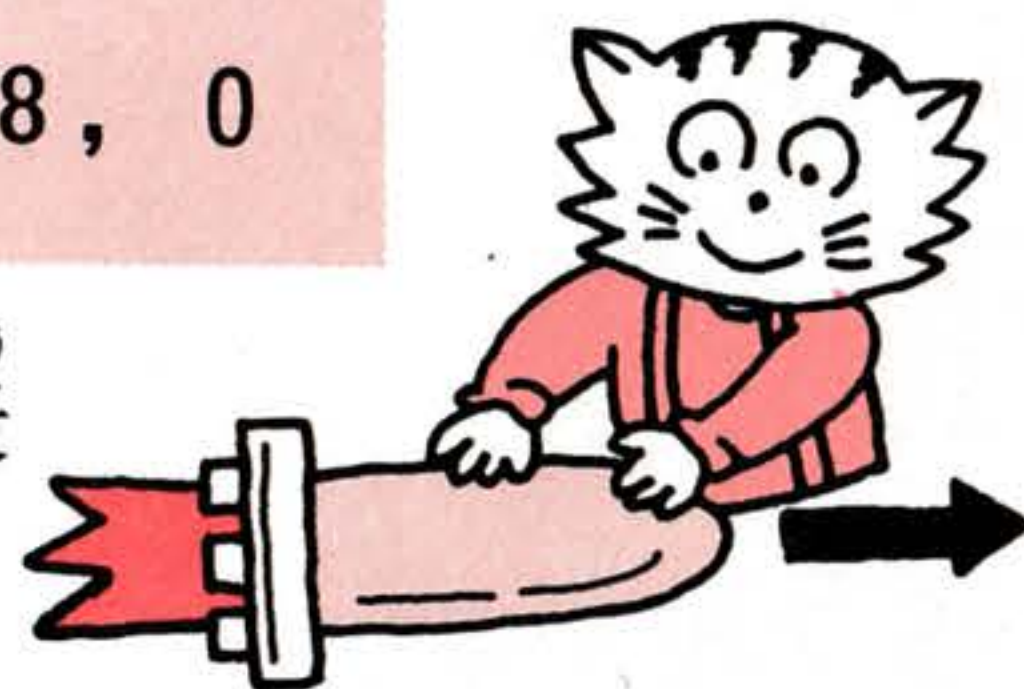
9 SCREEN 1, 0



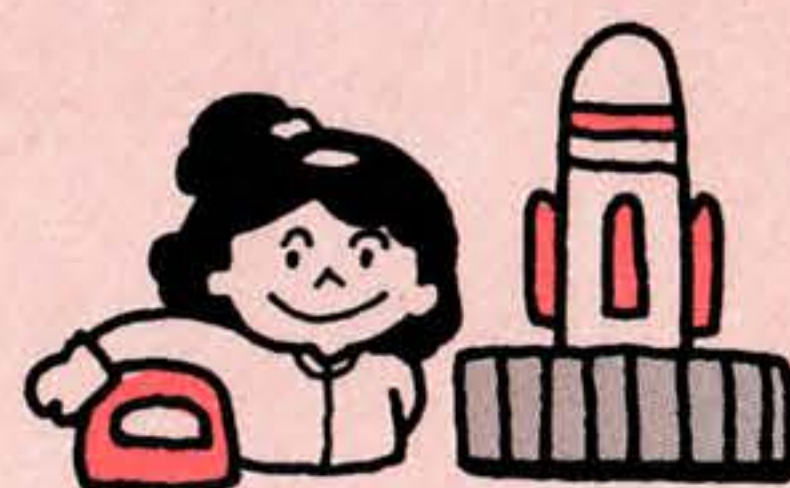
次に、いよいよPUT SPRITEで
画面に表示させるぞ。座標(0, 100)
に、赤(8)で表示させてみよう

110 PUT SPRITE 0, (0, 100), 8, 0

画面に出ただけじゃつまらない。スプライトの表
示場所を変えれば、左から右に動くようになるぞ。



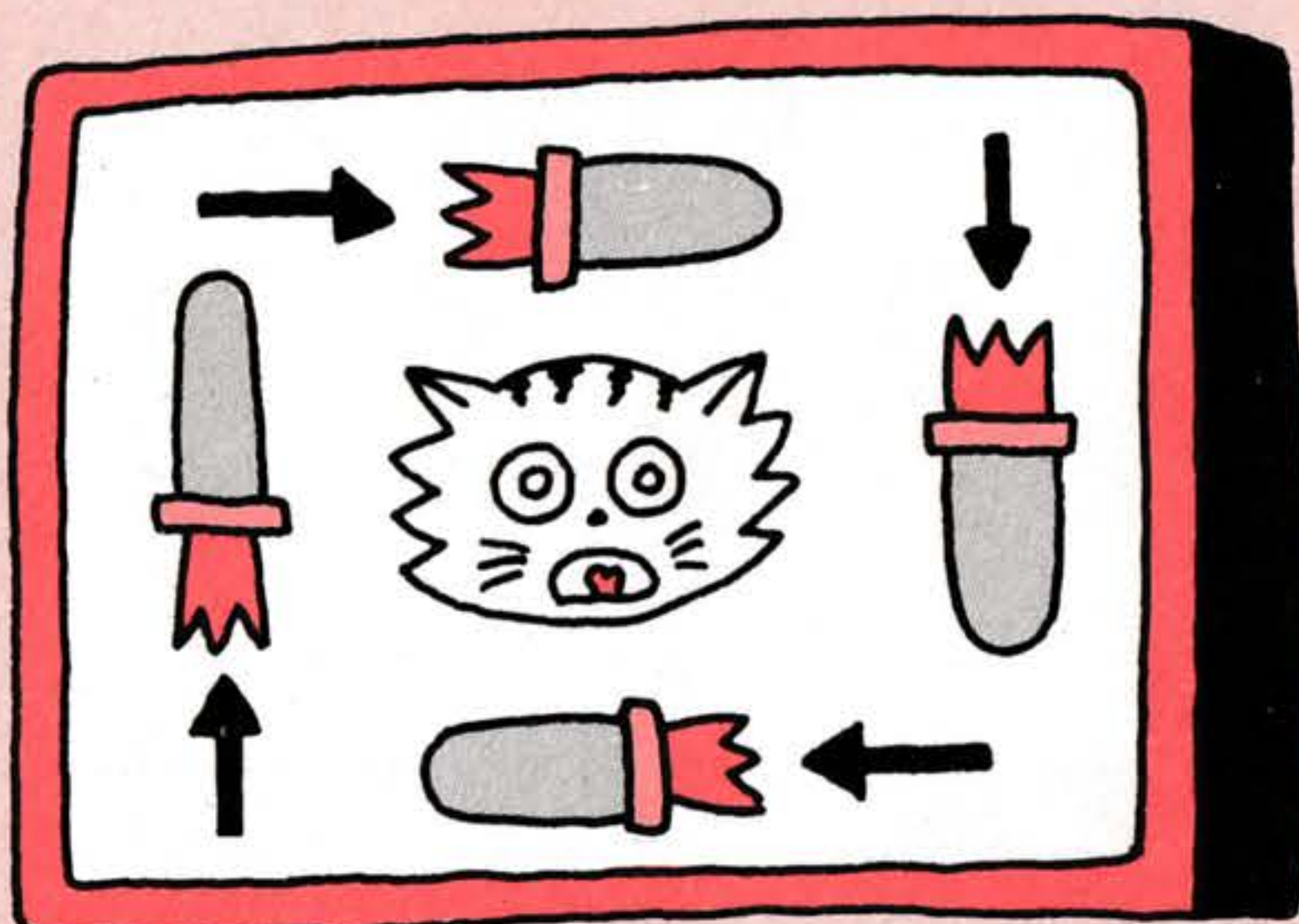
```
10 SCREEN 1, 0
20 S1$=CHR$(&B000001000)
30 S2$=CHR$(&B000001000)
40 S3$=CHR$(&B000011100)
50 S4$=CHR$(&B000011100)
60 S5$=CHR$(&B01011101)
70 S6$=CHR$(&B01111111)
80 S7$=CHR$(&B01111111)
90 S8$=CHR$(&B01011101)
100 SPRITE$(0)=S1$+S2$+S3$+S4$+S5$+S6$+S7$+S8$
110 PUT SPRITE 0, (0, 100), 8, 0
120 FOR N=1 TO 230 STEP 2
130 PUT SPRITE 0, (N, 100), 8, 0
140 NEXT
150 COLOR 15, 4, 7
```





ロケットを上下左右に動かそう

```
10 SCREEN 1,0:KEY OFF
20 FOR I=0 TO 3
30 SP$=""
40 FOR J=0 TO 7
50 READ A:A$=CHR$(A)
60 SP$=SP$+A$
70 NEXT J
80 SPRITE$(I)=SP$
90 NEXT I
100 X=120:Y=88
110 A=STICK(0)
120 IF A=1 AND Y<>0 THEN Y=Y-8:SP=0
130 IF A=3 AND X<>240 THEN X=X+8:SP=1
140 IF A=5 AND Y<>184 THEN Y=Y+8:SP=3
150 IF A=7 AND X<>8 THEN X=X-8:SP=2
160 PUT SPRITE 0,(X,Y),8,SP
170 GOTO 110
180 DATA &H8,&H8,&H1C,&H1C,&H5D,&H7F,&H7F,&H5D
190 DATA &HFO,&H60,&HFC,&HFF,&HFC,&H60,&HFO,0
200 DATA &HF,&H6,&H3F,&HFF,&H3F,&H6,&HF,0
210 DATA &H5D,&H7F,&H7F,&H5D,&H1C,&H1C,&H8,&H8
```

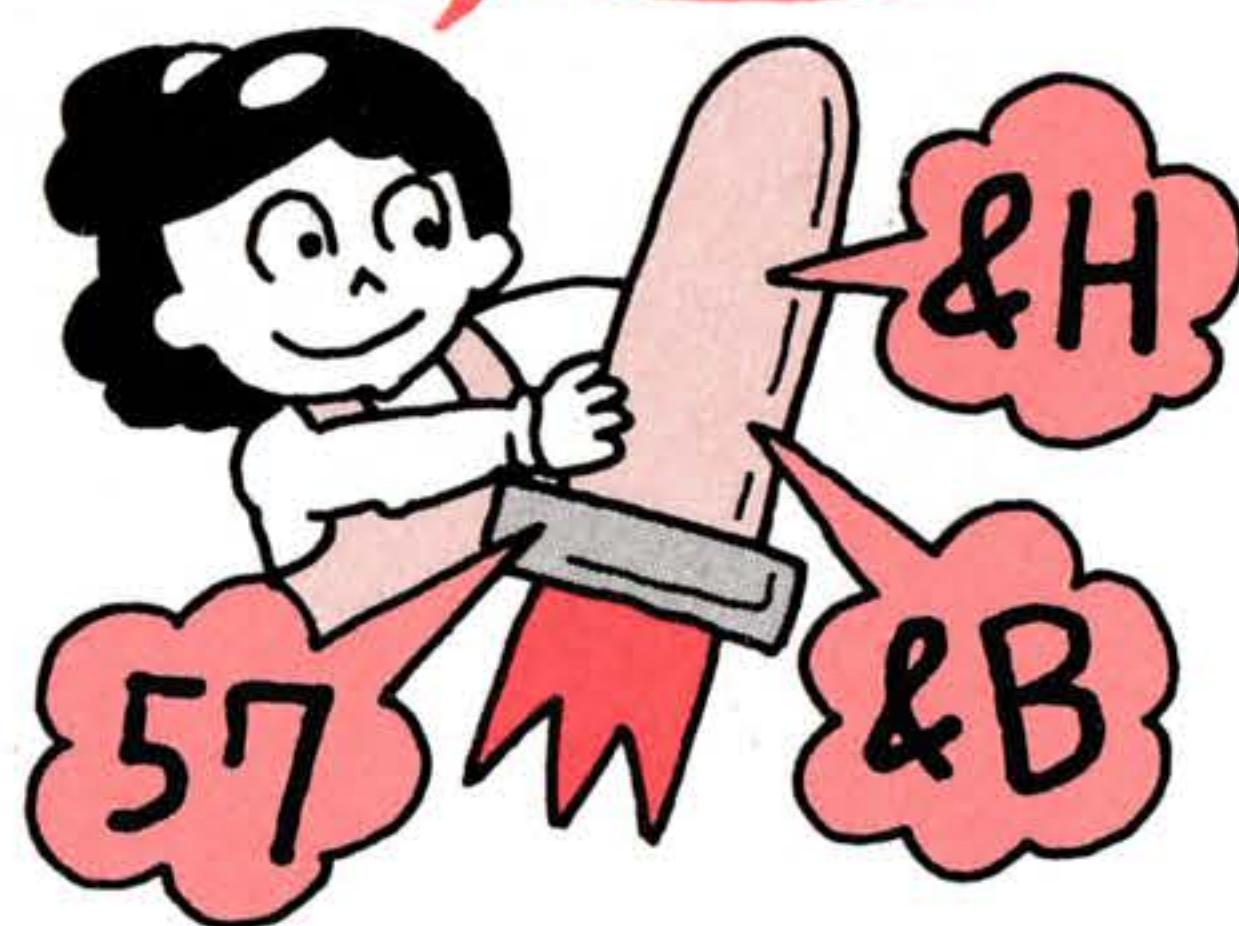


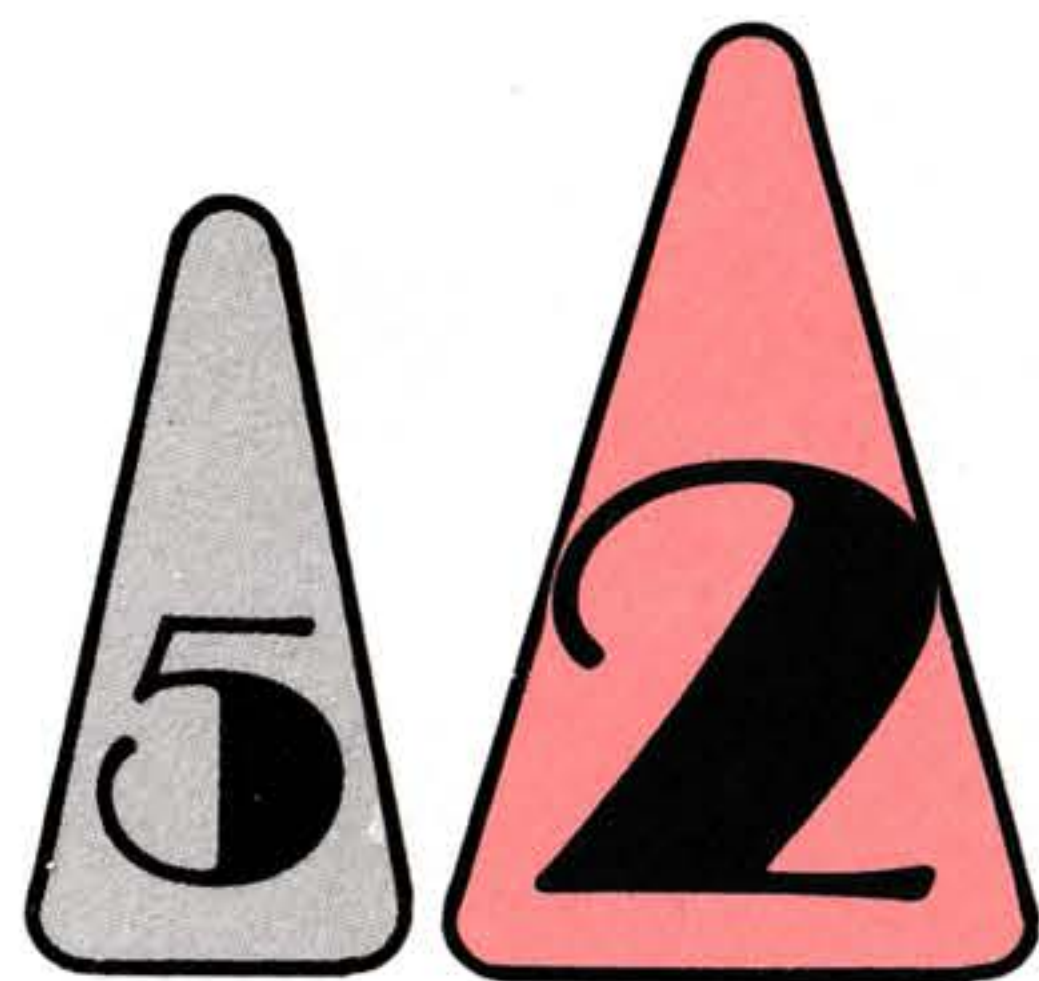
カーソルキーを押した^お方向^{ほうこう}に、ロケットが^{うご}動くプログラムだ。

まず、画面^{がめん}の中央^{ちゅうおう}にスプライト^{ひょうじ}で表示させたロケットがあらわれる。そのあと、上下左右^{じょうげさゆう}のカーソルキーを押してみよう。方向^{ほうこう}によって、ロケットの向き^むも^か変わるよ。

このプログラムでは、16^{しんすう}進数でスプライトパターンを定義^{ていぎ}してある。180^{ぎょう}行から210^{ぎょう}行のDATA文^{ぶん}が、4つの向き^むのスプライトパターンだよ。

180^{ぎょう}行から210^{ぎょう}行のデータを、さっきの2^{しんすう}進数^かに変えても同じ^{おは}パターンが出るよ





16×16の インベーダーだ



8×8が4つ集まったと考えよう

8×8ドットのパターンの表示では、縦に8個、横に8個、計64個のマス目があった。

16×16ドットのパターンなら、縦に16個、横に16個の合計256個のマス目があることになる。これだけマス目が多ければ、かんたんな8×8のパターンよりも、もっとおもしろいパターンも表示できるぞ。

ただし、注意しなければならないことがある。16×16のパターンを表示させるときは、SCREEN 1は、使えないよ。SCREEN 2または3のグラフィック・モードを使わないと表示できないからね。

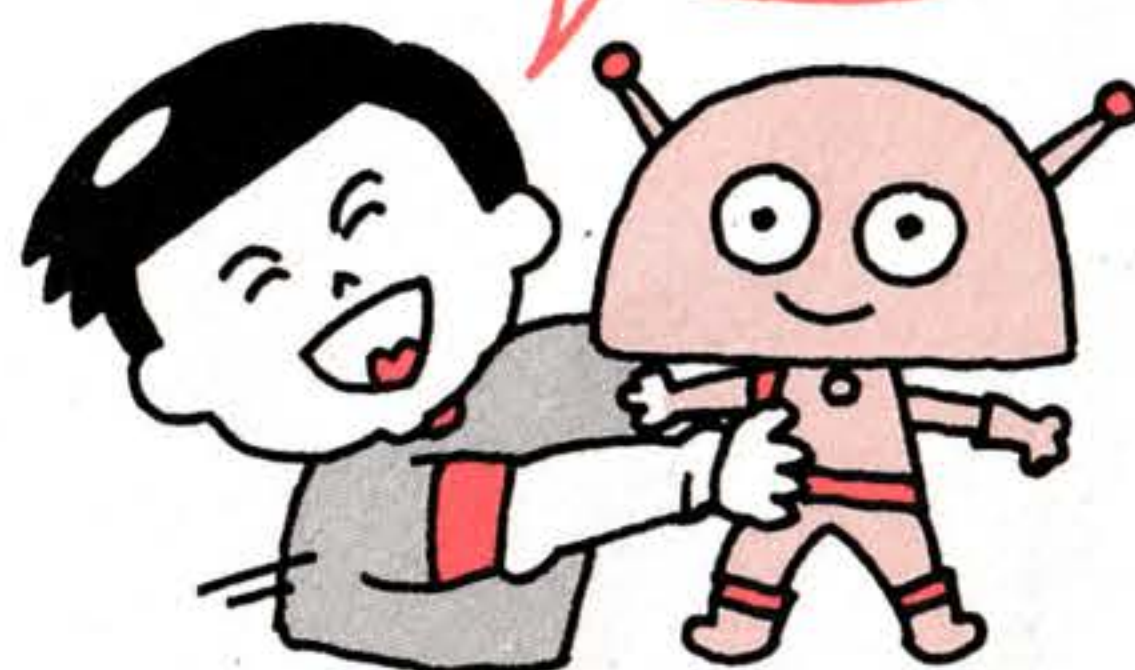
それともうひとつ、8×8のパターンは、最高256個まで用意できたけれど、16×16では、最高64個までだよ。つまり

SPRITES ()

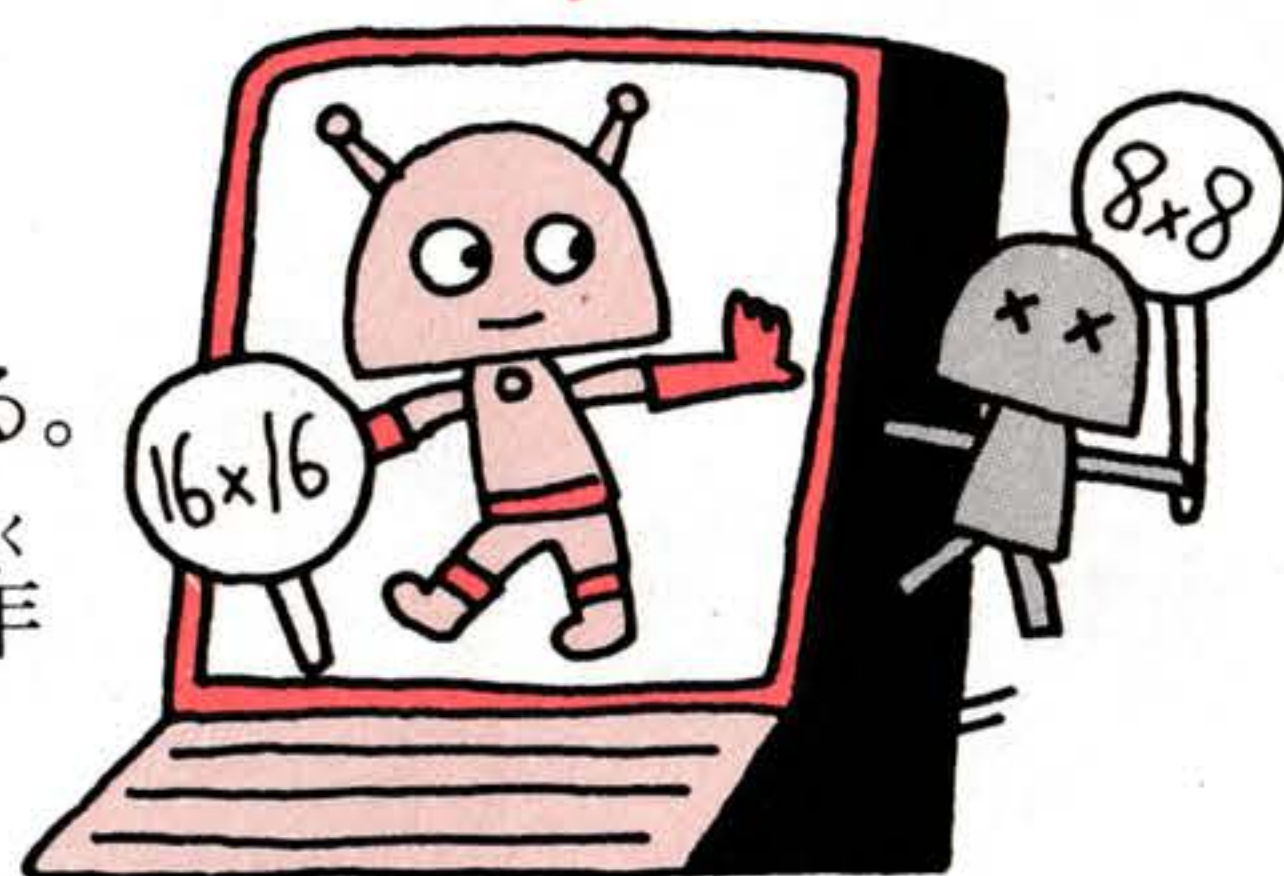
のカッコの中は、0から63までと決められている。

さあ、ここでは、インベーダーのような形を作ってみたぞ。

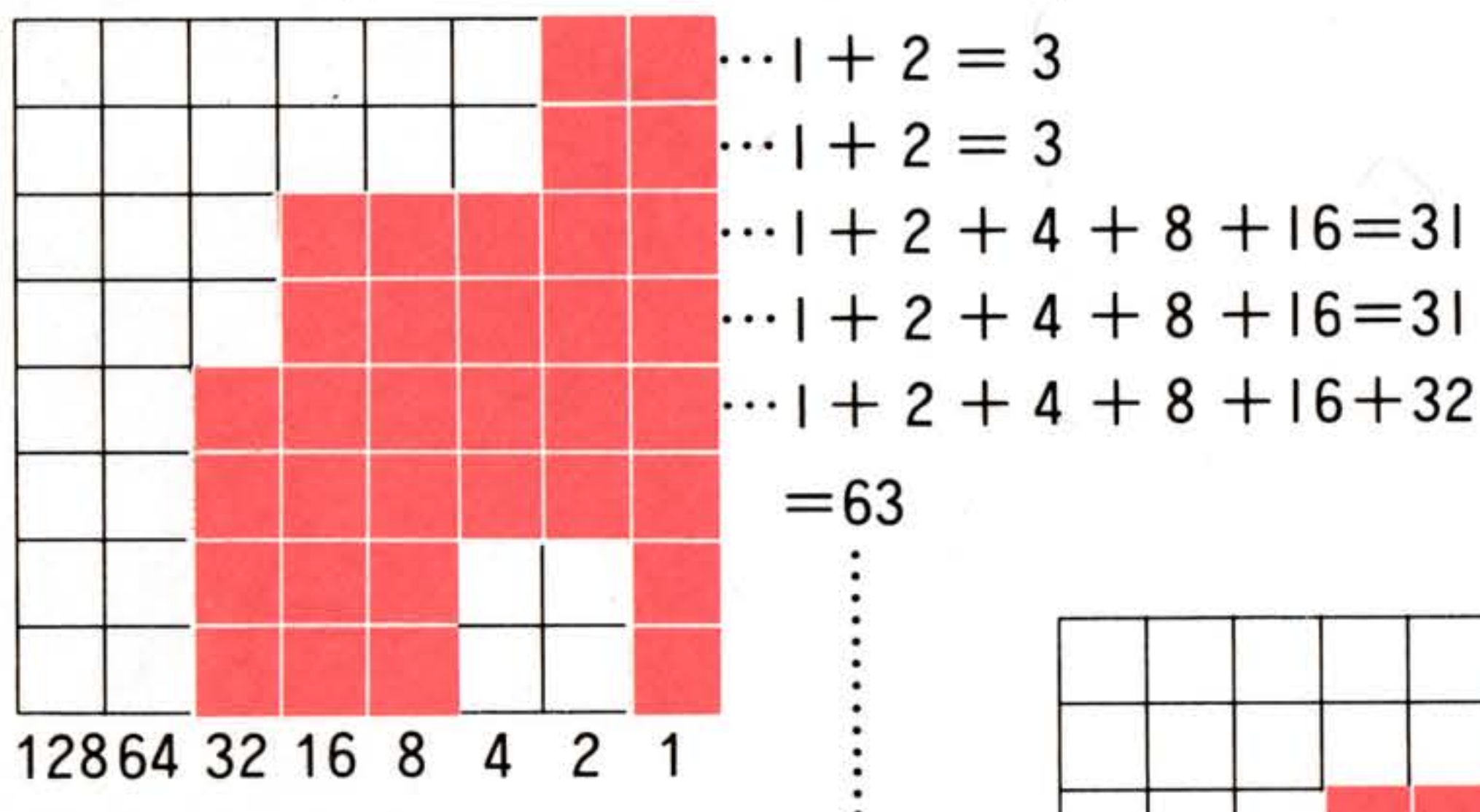
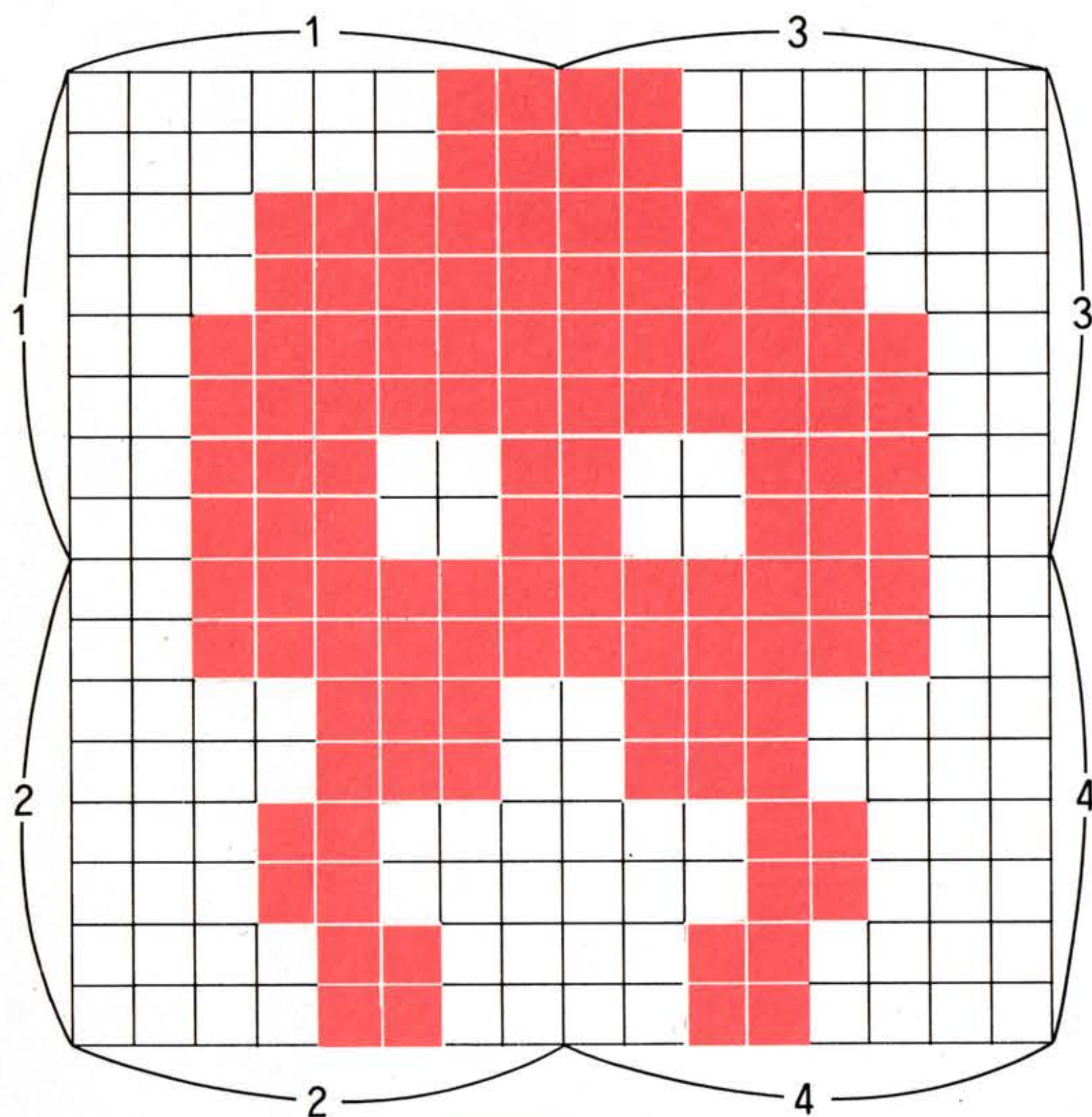
16×16ドットなら、もっと複雑なキャラクターも表示できて、おもしろさもグーンとアップだ



同じディスプレイ画面で、8×8のパターンと16×16のパターンを同時に表示させることはできないよ



●16×16のパターンを定義する



塗りつぶしたマス目の部分
 だけ下の数をたしていくと、
 10進数がでるよ。このデー
 タを4つ分出すんだ

$$1 \times 8 + 16 \times 3 = 56$$

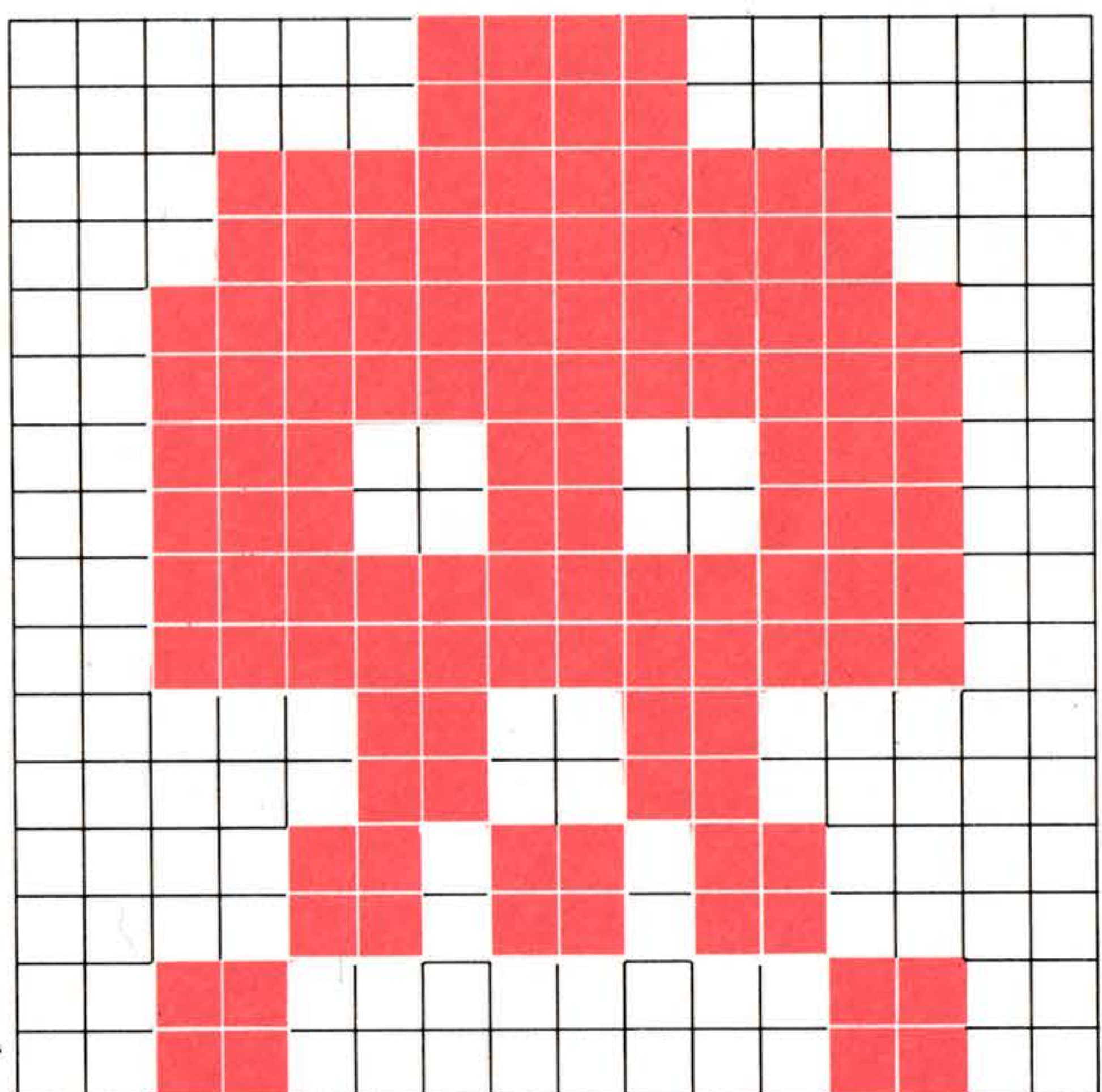


16×16ドットは、8×8
 ドットの大きなマスが4つ
 あつまったと考えよう。

縦横16個のマスを左右
 に2つ、上下に2つに分け
 ればいいんだ。

パターンを定義する順番
 は、左上が一番先だよ。

これだけのマスを2進
 数であらわしていたら、た
 いへんだ。メモリのむだ使
 いにもなる。そこで、こん
 どは10進数でパターンを定
 義してみよう。

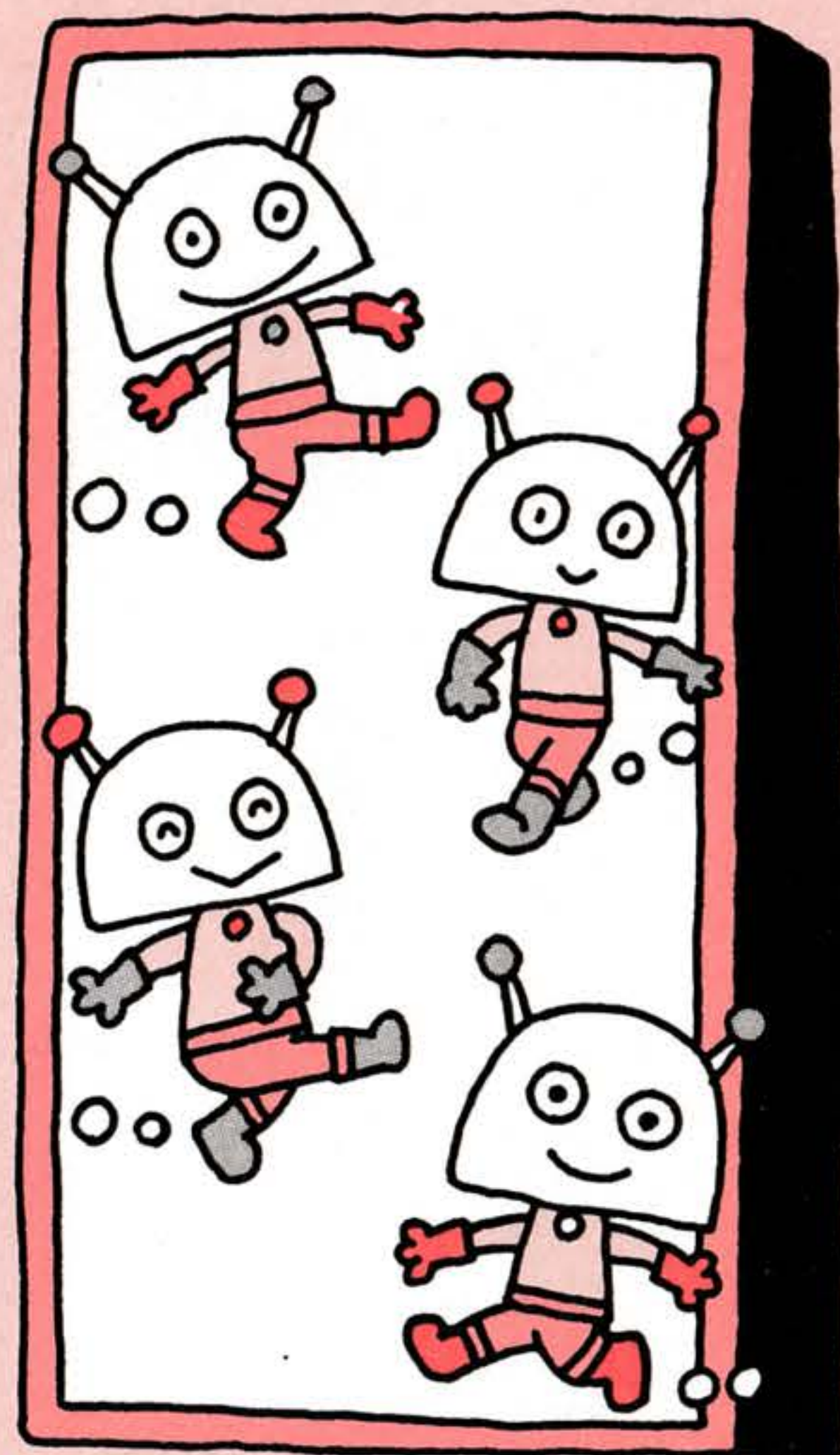


●インベーダーが左から右へ歩くプログラム

```

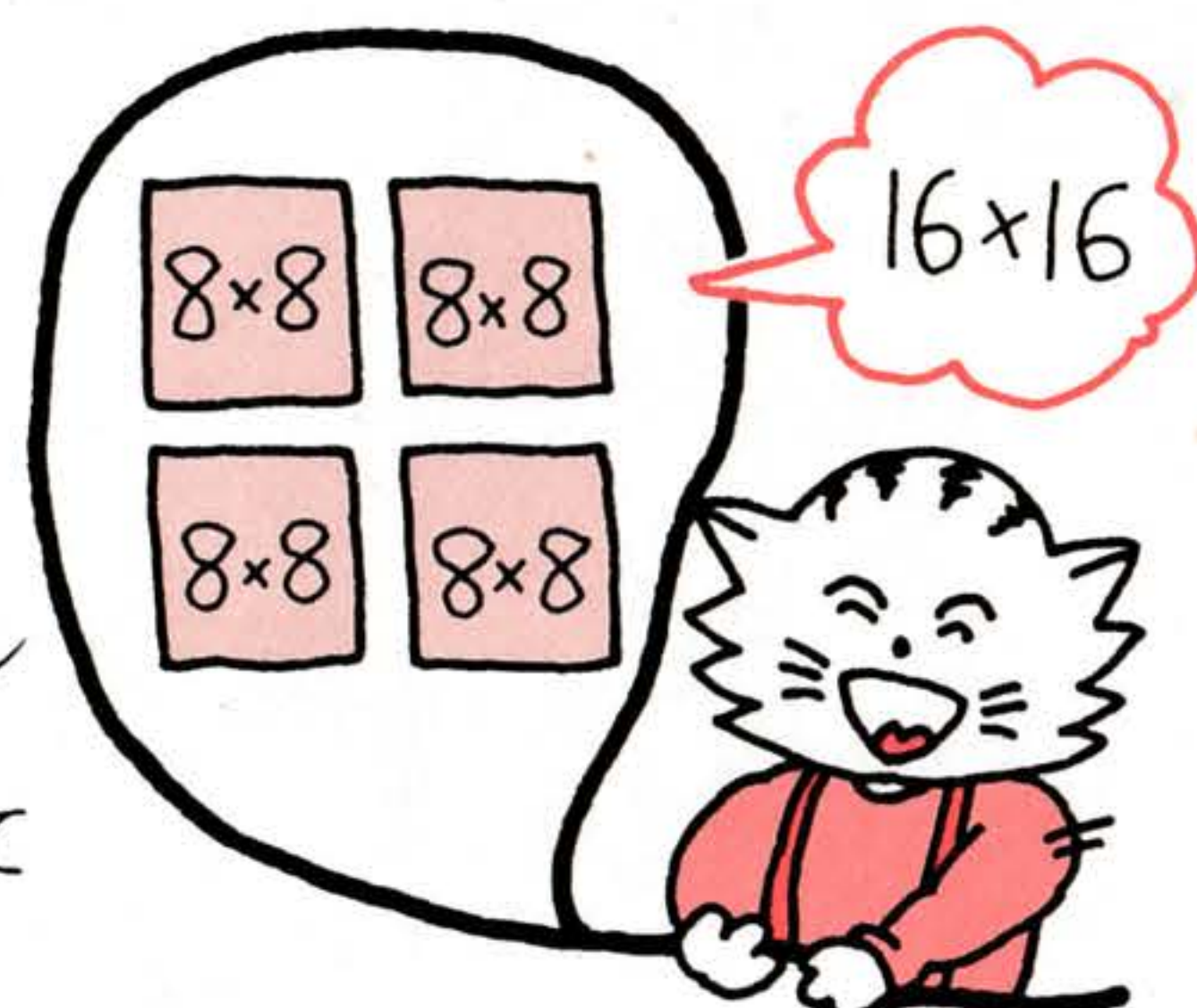
10 SCREEN 1,2
20 FOR I=1 TO2
30 SP$=""
40 FOR J=1 TO32
50 READ A:A$=CHR$(A)
60 SP$=SP$+A$
70 NEXT J
80 SPRITE$(I)=SP$
90 NEXT I
100 FOR K=16 TO232 STEP 16
110 PUT SPRITE 1,(K,100),10,1
120 GOSUB 170
130 PUT SPRITE 1,(K+8,100),10,2
140 GOSUB 170
150 NEXT K
160 END
170 PLAY "L2003c"
180 FOR L=1 TO300:NEXT L
190 RETURN
200 DATA &h3,&h3,&h1f,&h1f,&h3f,&h3f,&h39,&h39
210 DATA &h3f,&h3f,&h6,&h6,&hd,&hd,&h30,&h30
220 DATA &hc0,&hc0,&hf8,&hf8,&hfc,&hfc,&h9c,&h9c
230 DATA &hfc,&hfc,&h60,&h60,&hb0,&hb0,&hc,&hc
240 DATA &H3,&H3,&H1f,&H1f,&H3f,&H3f,&H39,&H39
250 DATA &h3f,&h3f,&he,&he,&h18,&h18,&hc,&hc
260 DATA &hc0,&hc0,&hf8,&hf8,&hfc,&hfc,&h9c,&h9c
270 DATA &hfc,&hfc,&h70,&h70,&h18,&h18,&h30,&h30

```



10進数の値は出せたかな？ 1段目から順に、
3、3、31、31、63、63、57、57になったかな。
白い部分は、たしちゃだめだよ。

上のプログラムは、左ページの2つのパターン
を交たいに表示させて、画面を左から右へ歩いて
いるように見せたプログラムだよ。



5 3 3次元迷路脱出ゲーム

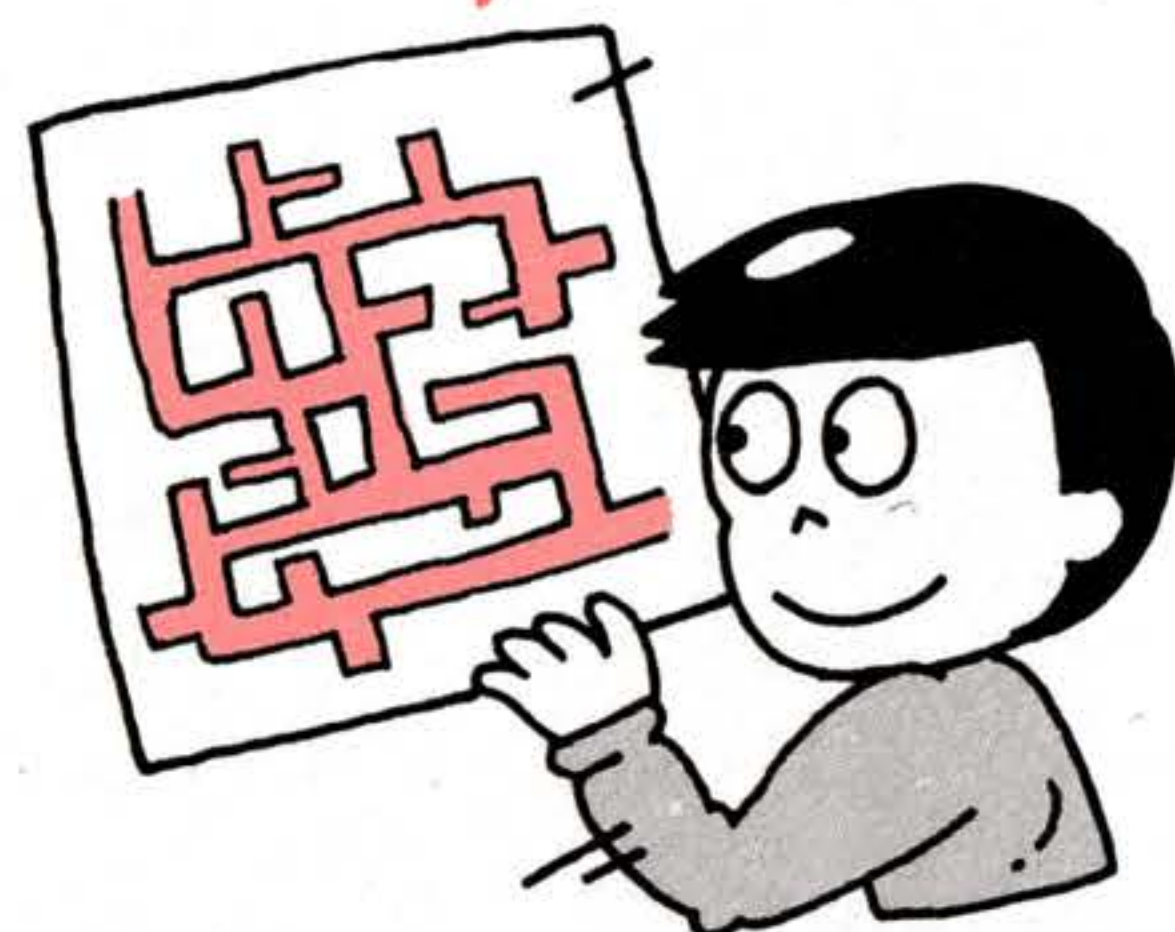


めい ろ へいめん ず あたま い
迷路の平面図を頭に入れてから

RUNすると、コンピュータが迷路を自動的に作成する。1分ぐらいたつと、画面に迷路の平面図が登場する。しばらくすると、この図は消えてしまうから、それまでに、どういう順序で進んで行ったら、迷路から脱出できるのかをしっかりと頭にたたき込んでおくのだ。

その後が、実際のゲームだ。きみがスタート位置に立ったとしよう。立体図がどんどんきみに迫ってくるから、キー入力しながらゴールをめざせ。スタートは、LOCATE (18, 18) だ。Yの値が少なくなるにつれて上に、Xの値が少なくなるにつれて左に移動していることになるぞ。

む ほうこう
向かっている方向を
しめ やじろし み
示す矢印を見ながら、
U.....1 歩前進
H.....90° 左へ変更
K.....90° 右へ変更
M.....うしろを向く
どんだんキー入力だ



● ゲームプログラム 3次元迷路からの脱出

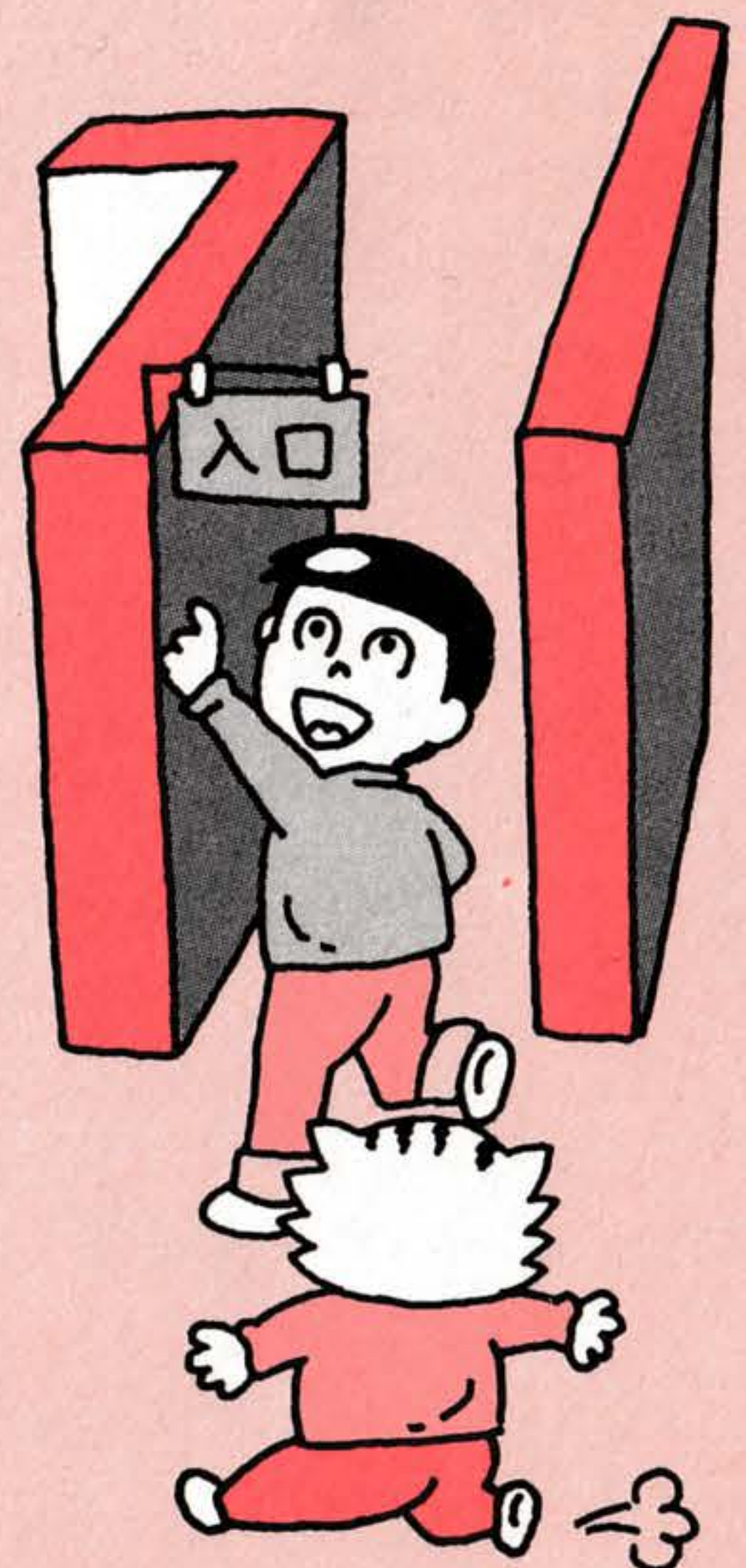
```
10 DIM M$(22,22),W$(4,2)
20 SCREEN 0
30 CLS
40 WIDTH 40:COLOR 10,1
50 LOCATE 10,2:PRINT"*** 3D迷路 ***"
60 LOCATE 5,5
70 PRINT"リッタイ ヒョウシ" サレル メイロ ラ ミナカラ,"
```



```

80 LOCATE 17,7
90 PRINT"メロヲヌケテワタサイ。"
100 LOCATE 10,10:PRINT"(U)KEY...マ イ"
110 LOCATE 10,12:PRINT"(H)KEY...ヒタリ"
120 LOCATE 10,14:PRINT"(K)KEY...ミ キ"
130 LOCATE 10,16:PRINT"(M)KEY...ウシロ"
140 LOCATE 10,18
150 LOCATE 14,21:PRINT"GET KEY"
160 K$=INKEY$:IF K$="" THEN 160
170 GOSUB 660
180 GOSUB 1880
190 SCREEN 2:OPEN"GRF:"AS#1
200 X=20:Y=20:HX=0:HY=-1
210 TIME=0
220 GOSUB 310
230 K$=INKEY$:IF K$="" THEN 230
240 IF K$="k" THEN GOSUB 610:GOTO 290
250 IF K$="h" THEN GOSUB 640:GOTO 290
260 IF K$="m" THEN HX=-HX:HY=-HY:GOTO 290
270 IF K$="u" THEN GOSUB 540:GOTO 290
280 GOTO 230
290 IF (X=2)AND(Y=2) THEN 460
300 GOTO 220
310 REM
320 CLS
330 GOSUB 1910
340 GOSUB 2130
350 FOR I=0 TO 4
360 PRESET(20*8,5*8-I*8)
370 FOR J=0 TO 2
380 PRINT#1,W$(I,J);
390 NEXT J,I
400 PRESET(21*8,5*8):PRINT#1,"!"
410 PRESET(17*8,9*8)
420 PRINT#1,"ホワイ ";:GOSUB 2900
430 PRESET(17*8,10*8)
440 PRINT#1,"イチ ";X-2;" ",Y-2
450 RETURN
460 REM
470 GOSUB 310
480 PRESET(17*8,14*8):PRINT#1,"タッシュツ!"

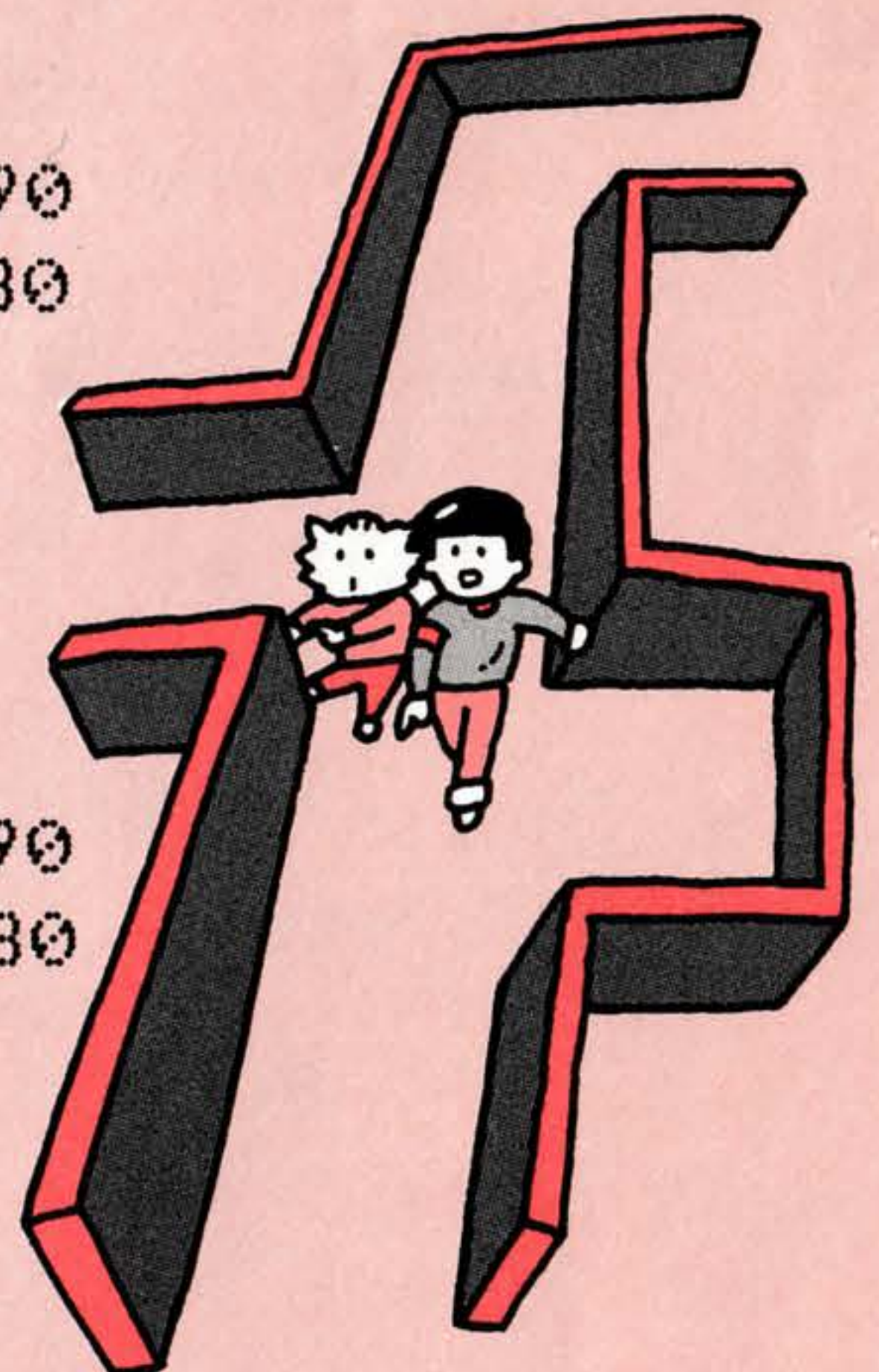
```




```

900 LOCATE 21,I:PRINT"♥♥"
910 NEXTI
920 FOR I=2 TO 20 STEP 2
930 FOR J=2 TO 20 STEP 2
940 IF M$(I,J)=" " THEN 1530
950 A1$=M$(I-2,J)
960 A2$=M$(I+2,J)
970 A3$=M$(I,J-2)
980 A4$=M$(I,J+2)
990 IF A1$+A2$+A3$+A4$="      " THEN 1530
1000 II=I:JJ=J
1010 M$(II,JJ)=" ":LOCATE II,JJ:PRINT" "
1020 IF M$(II,JJ-1)="♥" THEN 1050
1030 M$(II,JJ-1)=" "
1040 LOCATE II,JJ-1:PRINT" "
1050 IF JJ<>2 THEN 1090
1060 IF M$(II-1,JJ)="♥" THEN 1090
1070 M$(II-1,JJ)=" "
1080 LOCATE II-1,JJ:PRINT" "
1090 ONINT(RND(-TIME)*4)+1GOTO1100,1170,1240,1310
1100 IF M$(II-2,JJ)="♥" THEN 1090
1110 IF M$(II-2,JJ)=" " THEN 1380
1120 M$(II-1,JJ)=" "
1130 LOCATE II-1,JJ:PRINT" "
1140 M$(II-2,JJ)=" "
1150 LOCATE II-2,JJ:PRINT" "
1160 II=II-2:GOTO 1380
1170 IF M$(II+2,JJ)="♥" THEN 1090
1180 IF M$(II+2,JJ)=" " THEN 1380
1190 M$(II+1,JJ)=" "
1200 LOCATE II+1,JJ:PRINT" "
1210 M$(II+2,JJ)=" "
1220 LOCATE II+2,JJ:PRINT" "
1230 II=II+2:GOTO 1380
1240 IF M$(II,JJ-2)="♥" THEN 1090
1250 IF M$(II,JJ-2)=" " THEN 1380
1260 M$(II,JJ-1)=" "
1270 LOCATE II,JJ-1:PRINT" "
1280 M$(II,JJ-2)=" "
1290 LOCATE II,JJ-2:PRINT" "
1300 JJ=JJ-2:GOTO 1380

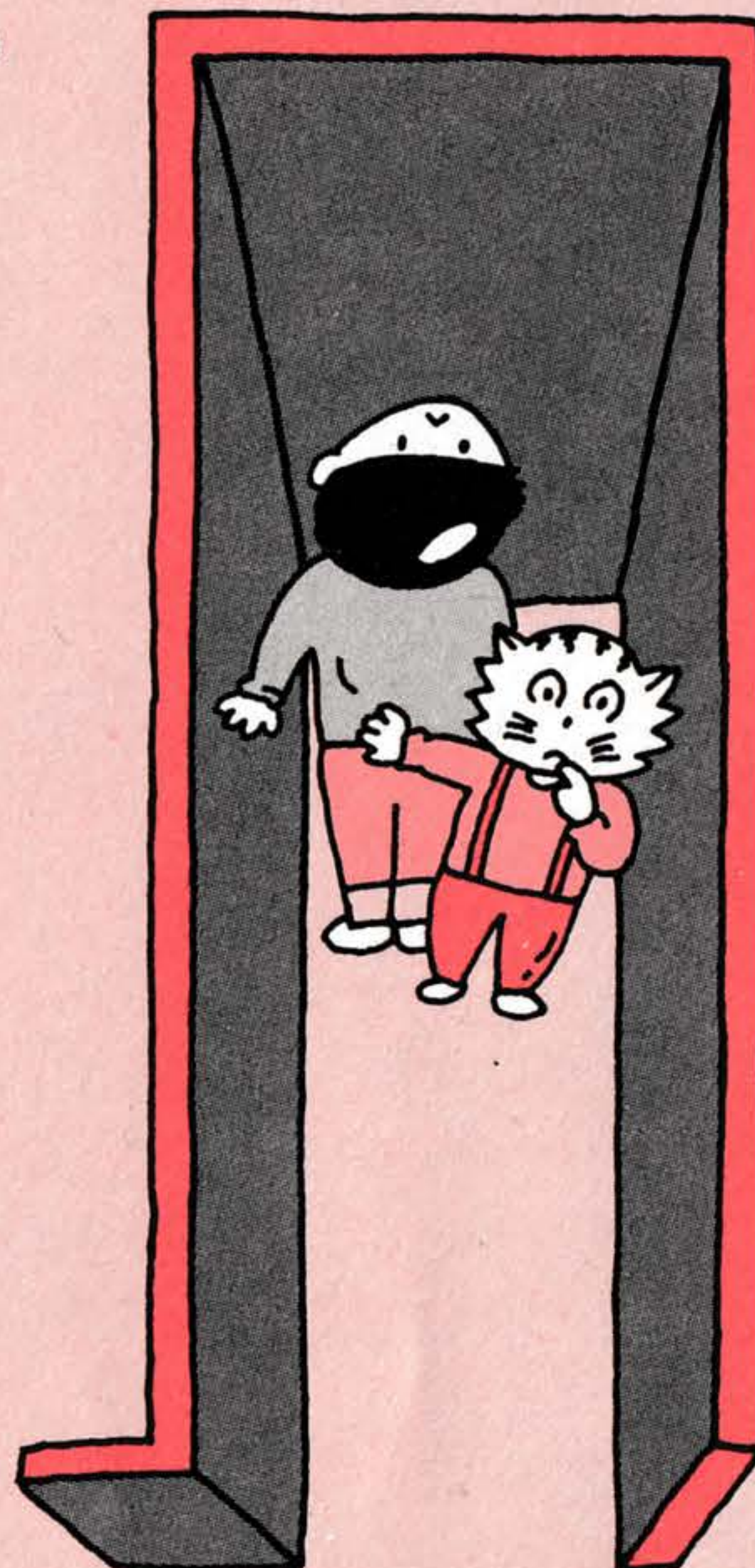
```




```

1310 IF M$(II,JJ+2)="♥" THEN 1090
1320 IF M$(II,JJ+2)=" " THEN 1380
1330 M$(II,JJ+1)=" "
1340 LOCATE II,JJ+1:PRINT" "
1350 M$(II,JJ+2)=" "
1360 LOCATE II,JJ+2:PRINT" "
1370 JJ=JJ+2:GOTO 1380
1380 B1$=M$(II-2,JJ)
1390 B2$=M$(II+2,JJ)
1400 B3$=M$(II,JJ-2)
1410 B4$=M$(II,JJ+2)
1420 B$=B1$+B2$+B3$+B4$
1430 IF B$=" " THEN 1530
1440 IF B$="♥" THEN 1530
1450 IF B$=" " ♥ " THEN 1530
1460 IF B$=" " ♥ " THEN 1530
1470 IF B$=" " ♥ " THEN 1530
1480 IF B$="♥ ♥ " THEN 1530
1490 IF B$="♥♥ " THEN 1530
1500 IF B$="♥ ♥ " THEN 1530
1510 IF B$="♥ ♥ " THEN 1530
1520 GOTO 1090
1530 NEXT J,I
1540 FOR I=2 TO 20 STEP 2
1550 FOR J=2 TO 20 STEP 2
1560 C1$=M$(I-2,J)
1570 C2$=M$(I+2,J)
1580 C3$=M$(I,J-2)
1590 C4$=M$(I,J+2)
1600 C$=C1$+C2$+C3$+C4$
1610 IF (C$=" ")AND(M$(I,J)="♥") THEN 1640
1620 NEXT J,I
1630 GOTO 1790
1640 M$(I,J)=" ":LOCATE I,J:PRINT" "
1650 ONINT(RND(-1)*4)+1 GOTO 1660,1700,1730,1760
1660 IF M$(I-1,J)="♥" THEN 1640
1670 M$(I-1,J)=" "
1680 LOCATE I-1,J:PRINT" "
1690 GOTO 1620
1700 IF M$(I+1,J)="♥" THEN 1640
1710 M$(I+1,J)=" ":LOCATE I+1,J:PRINT" "

```




```

1720 GOTO 1620
1730 IF M$(I,J-1)=" " THEN 1640
1740 M$(I,J-1)=" ":LOCATE I,J-1:PRINT" "
1750 GOTO 1620
1760 IF M$(I,J+1)="♥" THEN 1640
1770 M$(I,J+1)=" ":LOCATE I,J+1:PRINT" "
1780 GOTO 1620
1790 FOR K=0 TO 10
1800 LOCATE 2,1:PRINT"! "
1810 LOCATE 20,21:PRINT"! "
1820 FOR J=0 TO 500:NEXT J
1830 LOCATE 2,1:PRINT"♥"
1840 LOCATE 20,21:PRINT"♥"
1850 FOR J=0 TO 500:NEXT J
1860 NEXT K
1870 RETURN
1880 REM
1890 CLS
1900 RETURN
1910 REM
1920 FOR I=0 TO 4
1930 X1=X+HX*I:Y1=Y+HY*I
1940 IF (X1<0)OR(Y1<0)THEN W$(I,1)=" ":GOTO 1970
1950 IF(X1>22)OR(Y1>22)THENW$(I,1)=" ":GOTO 1970
1960 W$(I,1)=M$(X1,Y1)
1970 NEXT
1980 R1=HX*0+HY*1:R2=HX*-1+HY*0
1990 FOR I=0 TO 4
2000 X1=X+R1+HX*I:Y1=Y+R2+HY*I
2010 IF(X1<0)OR(Y1<0)THENW$(I,0)=" ":GOTO 2040
2020 IF(X1>22)OR(Y1>22)THENW$(I,0)=" ":GOTO 2040
2030 W$(I,0)=M$(X1,Y1)
2040 NEXT
2050 R1=HX*0+HY*-1:R2=HX*1+HY*0
2060 FOR I=0 TO 4
2070 X1=X+R1+HX*I:Y1=Y+R2+HY*I
2080 IF(X1<0)OR(Y1<0)THENW$(I,2)=" ":GOTO 2110
2090 IF(X1>22)OR(Y1>22)THENW$(I,2)=" ":GOTO 2110
2100 W$(I,2)=M$(X1,Y1)
2110 NEXT
2120 RETURN

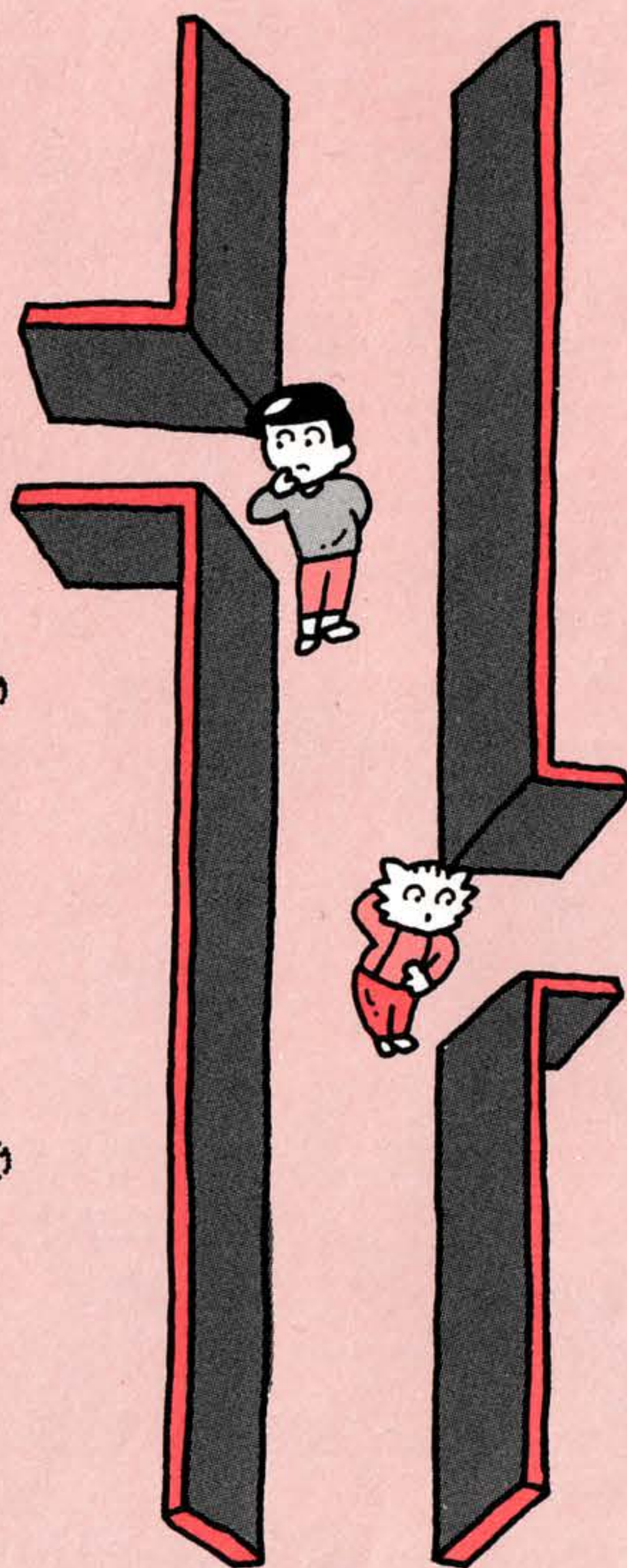
```




```

2130 REM
2140 FOR I=1 TO 4
2150 IF W$(I,1)="W" THEN 2180
2160 IF W$(I,1)="E" THEN 2180
2170 NEXT
2180 ON I GOTO 2280,2250,2220,2190,2330
2190 LINE (71,71)-(80,71):LINE-(80,80)
2200 LINE-(71,80):LINE-(71,71)
2210 GOTO 2330
2220 LINE (64,63)-(87,63)
2230 LINE (64,88)-(87,88)
2240 GOTO 2470
2250 LINE (48,47)-(103,47)
2260 LINE (48,104)-(103,104)
2270 GOTO 2590
2280 LINE (24,23)-(127,23)
2290 LINE(24,128)-(127,128)
2300 LINE (23,24)-(23,127)
2310 LINE (128,24)-(128,128)
2320 GOTO 2710
2330 IF W$(3,0)=" " THEN 2370
2340 LINE (64,64)-(71,71)
2350 LINE (64,87)-(71,80)
2360 GOTO 2400
2370 LINE (64,71)-(71,71)
2380 LINE(64,80)-(71,80)
2390 LINE (71,71)-(71,80)
2400 IF W$(3,2)=" " THEN 2440
2410 LINE (80,71)-(87,64)
2420 LINE (80,80)-(87,87)
2430 GOTO 2470
2440 LINE (80,71)-(87,71)
2450 LINE(80,80)-(87,80)
2460 LINE (80,71)-(80,80)
2470 IF W$(2,0)=" " THEN 2510
2480 LINE (48,48)-(63,63)
2490 LINE -(63,88):LINE -(48,103)
2500 GOTO 2530
2510 LINE (48,63)-(63,63)
2520 LINE -(63,88):LINE -(48,88)
2530 IF W$(2,2)=" " THEN 2570

```




```

2540 LINE (103,48)-(88,63)
2550 LINE -(88,88):LINE-(103,103)
2560 GOTO 2590
2570 LINE (103,63)-(88,63)
2580 LINE-(88,88):LINE-(103,88)
2590 IF W$(1,0)=" " THEN 2630
2600 LINE (24,24)-(47,47)
2610 LINE-(47,104):LINE-(24,127)
2620 GOTO 2650
2630 LINE (24,47)-(47,47)
2640 LINE-(47,104):LINE-(24,104)
2650 IF W$(1,2)=" " THEN 2690
2660 LINE (127,24)-(104,47)
2670 LINE-(104,104):LINE-(127,127)
2680 GOTO 2710
2690 LINE (127,47)-(104,47)
2700 LINE-(104,104):LINE-(127,104)
2710 LINE (23,24)-(23,127)
2720 LINE (128,24)-(128,127)
2730 IF W$(0,0)=" " THEN 2770
2740 LINE (16,16)-(23,23)
2750 LINE (16,135)-(23,128)
2760 GOTO 2790
2770 LINE (16,23)-(23,23)
2780 LINE (16,128)-(23,128)
2790 IF W$(0,2)=" " THEN 2830
2800 LINE (128,23)-(135,16)
2810 LINE (128,128)-(135,135)
2820 GOTO 2850
2830 LINE (128,23)-(135,23)
2840 LINE (128,128)-(135,128)
2850 PRESET(72,72)
2860 IF(X=2)AND(Y<5)AND(HY=-1)THEN PRINT#1,"G"
2870 PRESET(72,72)
2880 IF(X<5)AND(Y=2)AND(HX=-1) THEN PRINT#1,"G"
2890 RETURN
2900 IF (HX=0)AND(HY=1) THEN PRINT#1,"S";
2910 IF (HX=0)AND(HY=-1) THEN PRINT#1,"N";
2920 IF (HX=1)AND(HY=0) THEN PRINT#1,"E";
2930 IF (HX=-1)AND(HY=0) THEN PRINT#1,"W"
2940 RETURN

```





キーボード操作の トレーニング



ベーシック アルファベット&BASIC

●トレーニングのメニューは4つ

キーボードをしっかりと^{せいかく}正確に^お押すこと。これはいちばんたいせつなことだ。いっしょうけんめい^{にゅうりょく}入力した^{なが}長いプログラムも、1字^じまちがえてい^だるだけで、エラーを出して途中で止まってしまう。

そこで、キーボードを^{てっていき}徹底的にマスターするプログラムを作^{つく}ってみた。

何回^{なんかい}もRUNして^{れんしゅう}練習すれば、キーボードを^{はや}速く、^{せいかく}正確に打てるようになるはずだ。最後^{さいご}には、キーボードを見^みずに打てるくらい上達^{じょうたつ}したいね。

プログラムをRUNさせると、キーボードの^え絵と4つのトレーニングメニューが^で出てくる。きみが^{れんしゅう}練習したいと思^{おも}うメニューのところに、カーソル^{つか}を使って^{いどう}移動して、**RETURN** キーを^お押す。

カーソルは上向きと下向きの2つを使うんだよ。

トレーニングメニューは、次の4項目^{つぎ こうもく ようい}を用意した。

1. アルファベットの^{れんしゅう}練習

アルファベットのA～Zまでを^{じゅんばん}順番に^お押す^{れんしゅう}練習だ。画面^{がめん}のキーボードに^{あか}赤い^{わく}枠が^{ひょうじ}表示さ

1. アルファベットの^{れんしゅう}練習



2. アルファベットのテスト



れるから、その通りキーを押していく。

2. アルファベットのテスト

はじめにA～Zまでのアルファベットを押すのに何秒かかるかをテストする。そのあとで、ランダムに出てくるアルファベットを1分間に何回押せるかをテストする。これは、キーボードを見ずに入力する練習にもなる。

3. BASIC命令の入力練習

BASICの命令が画面に表示されるから、きみはその命令をキーボードから入力する。1のように、画面上のキーに赤い枠が表示される。

4. BASIC命令の入力テスト

画面に表示されるBASICの命令をすべて入力するのにどのくらいの時間がかかるかをテストする。

3. BASIC命令の

入力練習



4. BASIC命令の

入力テスト



●トレーニングプログラム キーボードマスター

```
10 DIM AD(26,1),CM$(200),CM(200)
20 COLOR 15,14
30 SCREEN 2,0
40 RESTORE 3580
50 FOR I=0 TO 25
60 READ A,B
70 AD(I,0)=A:AD(I,1)=B
80 NEXT
90 OPEN"grp:"FOR OUTPUT AS#1
100 GOSUB 660
110 SCREEN ,0
```




```

120 LINE(12,8)-(248,90),4,BF
130 CA$=""
140 RESTORE 3530
150 FOR I=1 TO 8
160 READ CA:CA$=CA$+CHR$(CA)
170 NEXT
180 SPRITE$(1)=CA$
190 FOR I=0 TO 4
200 CIRCLE (60,20+I*16),3,9
210 NEXT
220 COLOR 15,4
230 PRESET (80,16)
240 PRINT#1,"アルファベットノ レンショウ"
250 PRESET (80,32)
260 PRINT#1,"アルファベットノ テスト"
270 PRESET (80,48)
280 PRINT#1,"コメント`ニューリョクノ レンショウ"
290 PRESET (80,64)
300 PRINT#1,"コメント`ニューリョクノ テスト"
310 PRESET (80,80)
320 PRINT#1,"オワリ"
330 XD=0
340 A$=INKEY$:IF A$<>"" GOTO 340
350 PUT SPRITE 1,(57,17+XD*16),9,1
360 A$=INKEY$
370 IF A$=CHR$(30)ANDXD<>0 THEN XD=XD-1
380 IF A$=CHR$(31)ANDXD<>4 THEN XD=XD+1
390 IF A$=CHR$(13) AND XD=4 THEN END
400 IF A$=CHR$(13) GOTO 420
410 GOTO 350
420 SCREEN ,3
430 RESTORE 3540
440 CA$=""
450 FOR I=1 TO 8
460 READ A:CA$=CA$+CHR$(A)
470 NEXT
480 SPRITE$(2)=CA$
490 RESTORE 3550
500 CA$=""
510 FOR I=1 TO 24
520 READ A:CA$=CA$+CHR$(A)

```




```

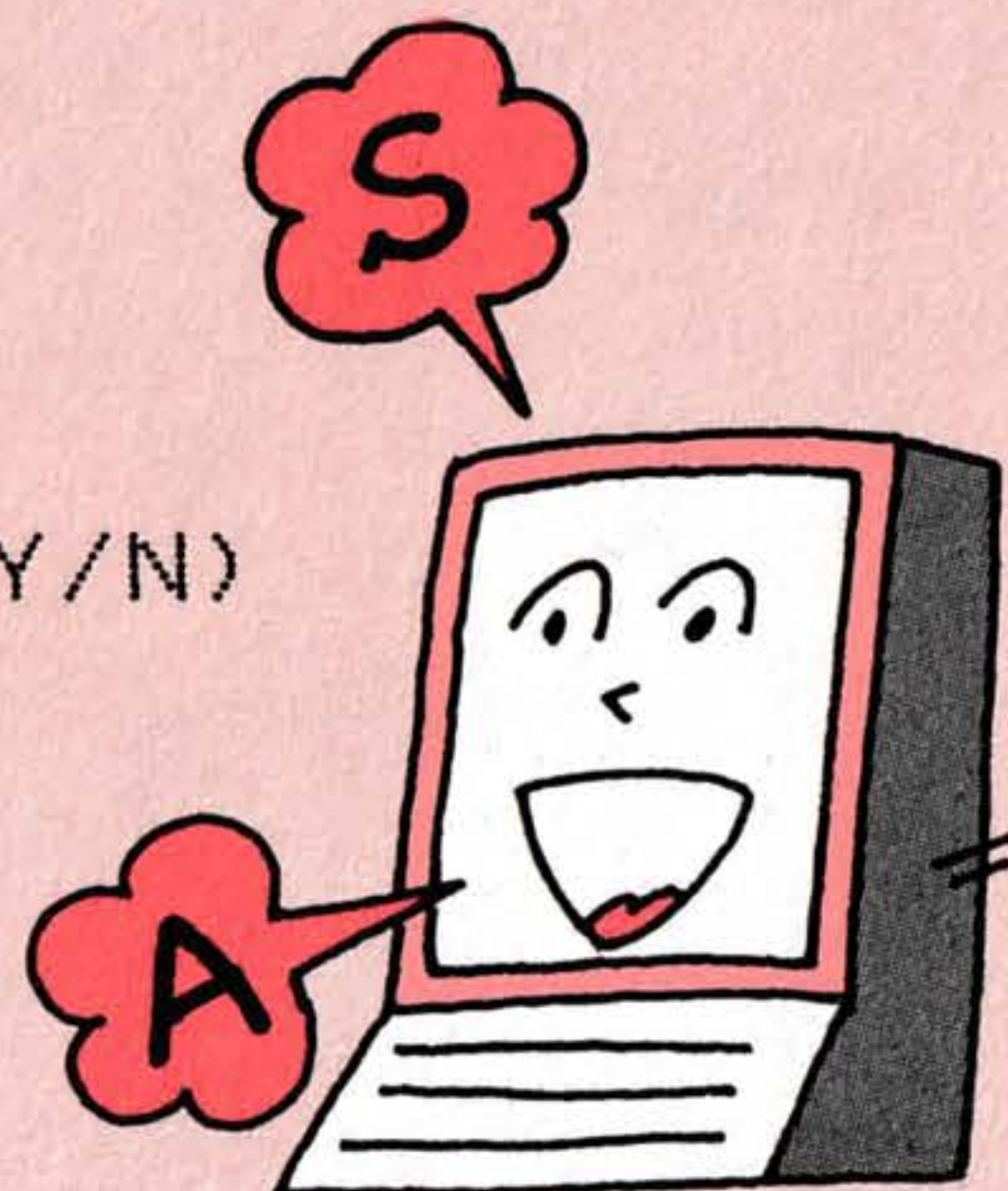
530 NEXT
540 SPRITE$(3)=CA$
550 LINE(12,8)-(248,90),4,BF
560 PRESET(24,16)
570 PRINT#1,"ハシ"メルマエニ CAPSLOCKキート カナキーカ"
580 PRESET(24,28)
590 PRINT#1,"オサレテイナイコトヲ カクニンシテワタ"サイ。"
600 PRESET (24,50)
610 PRINT#1,"カクニンシタラ スハ"ーヌキーヲ オシテワタ"サイ。"
620 A$=INKEY$
630 IF A$<>" " GOTO 620
640 ON XD+1 GOSUB 1010,1340,2150,2720
650 GOTO110
660 "キーボード" ヒョウシ"
670 LINE(12,8)-(248,90),4,BF
680 LINE(12,91)-(248,182),11,BF
690 FOR I=0 TO 12
700 LINE (I*16+24,100)-(I*16+36,112),15,BF
710 NEXT
720 FOR I=0 TO 11
730 LINE (I*16+32,116)-(I*16+44,128),15,BF
740 NEXT
750 FOR I=0 TO 11
760 LINE (I*16+40,132)-(I*16+52,144),15,BF
770 NEXT
780 LINE (24,148)-(44,160),15,BF
790 FOR I=0 TO 11
800 LINE (I*16+48,148)-(I*16+60,160),15,BF
810 NEXT
820 LINE (228,148)-(244,160),15,BF
830 LINE (80,164)-(204,176),15,BF
840 DRAW"BM24,180C7"
850 DRAW"M+4,-16R4M+2,10M+2,-10R4M+3,12"
860 DRAW"R9U2L7U10R18M+3,5M+3,-5R5"
870 DRAW"M-5,8M+5,8L5M-3,-5M-3,5L5M+5,-8M-4,-4"
880 DRAW"L10D3R6D9L15M-2,-10M-2,10"
890 DRAW"L4M-2,-10M-2,10L5"
900 PAINT(25,178),7
910 COLOR 0
920 PRESET(27,102),15
930 PRINT#1,"1 2 3 4 5 6 7 8 9 0 - ^ ￥"
    
```



```

940 PRESET(35,118),15
950 PRINT#1,"Q W E R T Y U I O P @ ["
960 PRESET(43,134),15
970 PRINT#1,"A S D F G H J K L ; : ]"
980 PRESET(27,150),15
990 PRINT#1,"SH Z X C V B N M , . / SH"
1000 RETURN
1010 'アルファベットノ レンショウ
1020 LINE(12,8)-(248,90),4,BF
1030 PRESET (24,16)
1040 PRINT#1,"ヨ カラ ズマデ"ノアルファベットヲ
1050 PRESET (32,28)
1060 PRINT#1,"シ"ユンハ"ンニ オシテワタ"サイ。"
1070 PRESET (24,40)
1080 PRINT#1,"カ"ノシ"ョウノキーボ"ート"ニハ アカイワクカ"
1090 PRESET (32,52)
1100 PRINT#1,"ヒョウシ"サレマス。
1110 PRESET (24,64)
1120 PRINT#1,"ハシ"メルトキハ ス"ペースキーヲ オシテワタ"サイ。"
1130 A$=INKEY$
1140 IF A$="" GOTO 1150 ELSE 1130
1150 LINE(20,78)-(234,88),2,BF
1160 FOR I=0 TO 25
1170 PUT SPRITE 2,(AD(I,0),AD(I,1)),8,2
1180 A$=INKEY$:IF A$<>""GOTO 1180
1190 A$=INKEY$:IF A$="" GOTO 1190
1200 IF A$=CHR$(97+I) GOTO 1230
1210 PLAY"C"
1220 GOTO 1190
1230 BEEP
1240 PRESET (24+I*8,80),2
1250 PRINT#1,CHR$(I+97)
1260 NEXT
1270 LINE(12,8)-(248,77),4,BF
1280 PRESET (24,50)
1290 PRINT#1,"モウイチ" ヤリマスカ ? (Y/N)
1300 A$=INKEY$
1310 IF A$="y" GOTO 1010
1320 IF A$="n" THEN RETURN
1330 GOTO 1300
1340 'アルファベットノ テスト

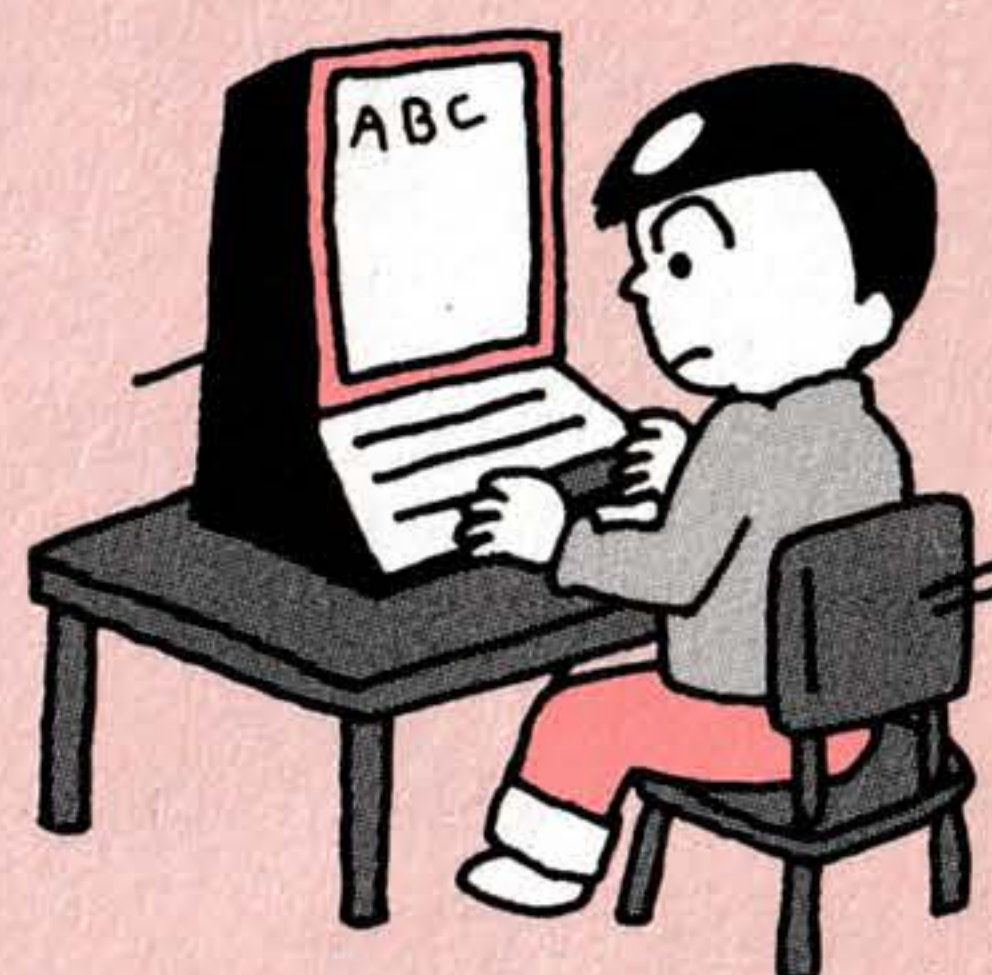
```




```

1350 LINE(12,8)-(248,90),4,BF
1360 PRESET (24,16)
1370 PRINT#1,"コノテストハ a カラ z マデノモシヲ
1380 PRESET (32,28)
1390 PRINT#1,"ナンヒョウデニユウリョクデキルカラ
1400 PRESET (32,40)
1410 PRINT#1,"タメテストデス。
1420 PRESET (24,52)
1430 PRINT#1,"スハースキーヲ オスト 3ヒョウコニ"
1440 PRESET (32,64)
1450 PRINT#1,"BEEPオンカナリ テストカハシマリマス。"
1460 A$=INKEY$
1470 IF A$<>" " GOTO 1460
1480 LINE(12,8)-(248,90),4,BF
1490 TIME=0
1500 IF TIME<180 GOTO 1500
1510 BEEP
1520 LINE(20,28)-(234,40),2,BF
1530 TIME=0
1540 FOR I=0 TO 25
1550 A$=INKEY$:IF A$<>" "GOTO 1550
1560 A$=INKEY$:IF A$="" GOTO 1560
1570 IF A$<"a" OR A$>"z" GOTO 1560
1580 AC=ASC(A$)-97
1590 PUT SPRITE 2,(AD(AC,0),AD(AC,1)),8,2
1600 IF A$=CHR$(97+I) GOTO 1630
1610 PLAY"C"
1620 GOTO 1560
1630 BEEP
1640 PRESET (24+I*8,30),2
1650 PRINT#1,CHR$(I+97)
1660 NEXT
1670 TI=TIME
1680 LINE(12,8)-(248,90),4,BF
1690 PRESET (40,16)
1700 PRINT#1,"タタイマノタイムハ";INT(TI/60);"ヒョウデシタ"
1710 PRESET (24,40)
1720 PRINT#1,"ツキハ ランタムニアラワレルアルファベットヲ"
1730 PRESET (32,52)
1740 PRINT#1,"17オンカンニ ナンモシニユウリョク"
1750 PRESET (32,64)

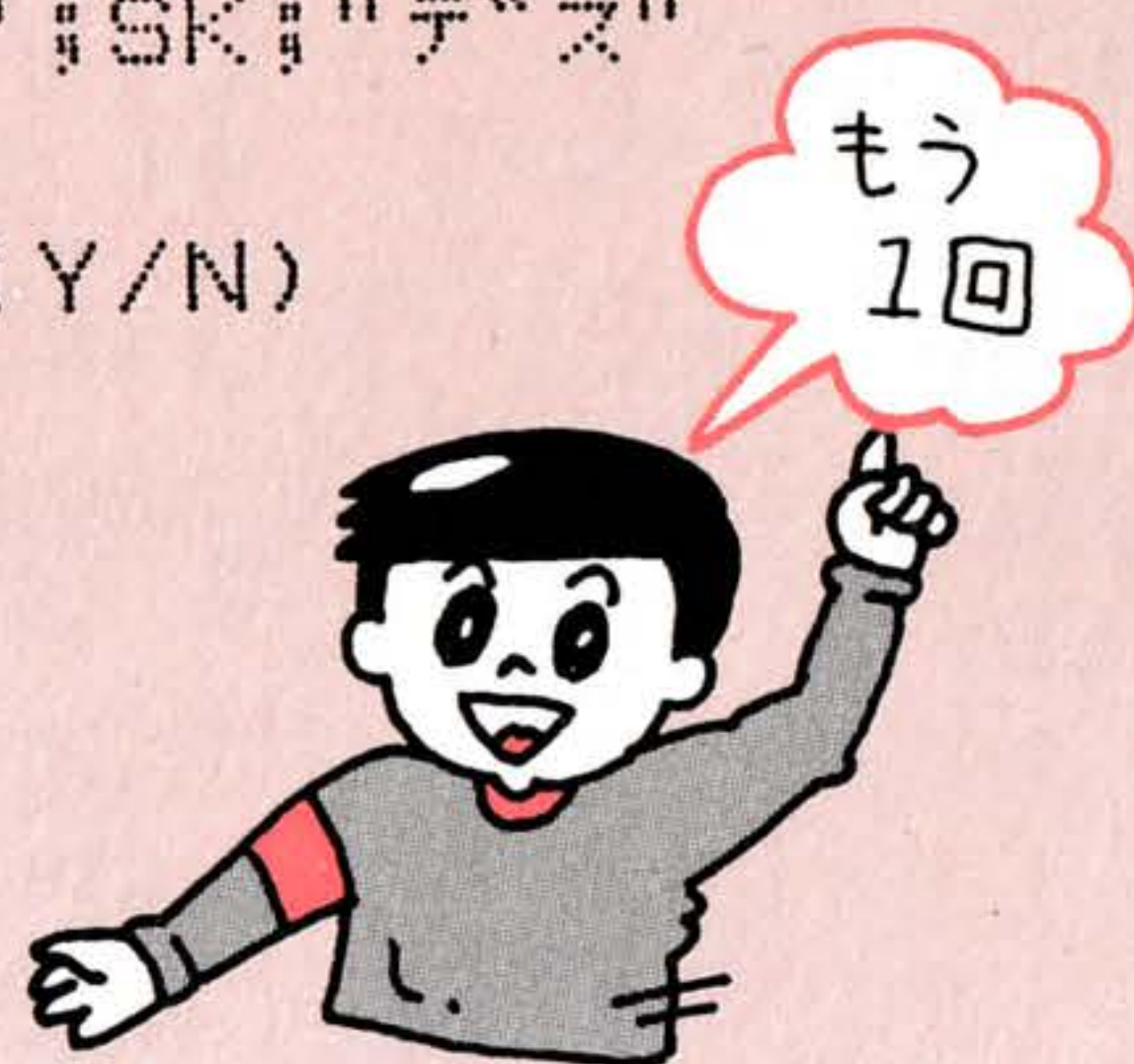
```




```

1760 PRINT#1,"テ"キルカラ タメテストテ"ズ。"
1770 PRESET (24,76)
1780 PRINT#1,"ス"ーキーヲ オストテストカ"ハシ"マリマス。"
1790 A$=INKEY$
1800 IF A$="" GOTO 1810 ELSE 1790
1810 LINE(12,8)-(248,90),4,BF
1820 PUT SPRITE 2,(0,209),0,0
1830 LINE (120,40)-(140,60),15,B
1840 FOR J=1 TO 500:NEXT J
1850 SK=0
1860 A=RND(-TIME)
1870 TIME=0
1880 RM=INT(RND(1)*26)+97
1890 PRESET (128,48)
1900 PRINT#1,CHR$(RM-32)
1910 A$=INKEY$
1920 IF TIME>3600 GOTO 2060
1930 IF A$<"a" OR A$>"z" GOTO 1910
1940 AC=ASC(A$)-97
1950 PUT SPRITE 2,(AD(AC,0),AD(AC,1)),8,2
1960 IF A$=CHR$(RM) THEN 1990
1970 PLAY"C"
1980 GOTO 1910
1990 BEEP
2000 SK=SK+1
2010 COLOR 4
2020 PRESET (128,48)
2030 PRINT#1,CHR$(RM-32)
2040 COLOR 15
2050 GOTO 1880
2060 LINE(12,8)-(248,90),4,BF
2070 PRESET (40,16)
2080 PRINT#1,"ニュウリョクシタモシ"ズウハ " ;SK;"テ"ズ"
2090 PRESET (24,40)
2100 PRINT#1,"モウイチト" マリマスカ ? (Y/N)
2110 A$=INKEY$
2120 IF A$="y" GOTO 1340
2130 IF A$="n" THEN RETURN
2140 GOTO 2110
2150 'コメント"ニュウリョクノ レンシュウ
2160 LINE(12,8)-(248,90),4,BF

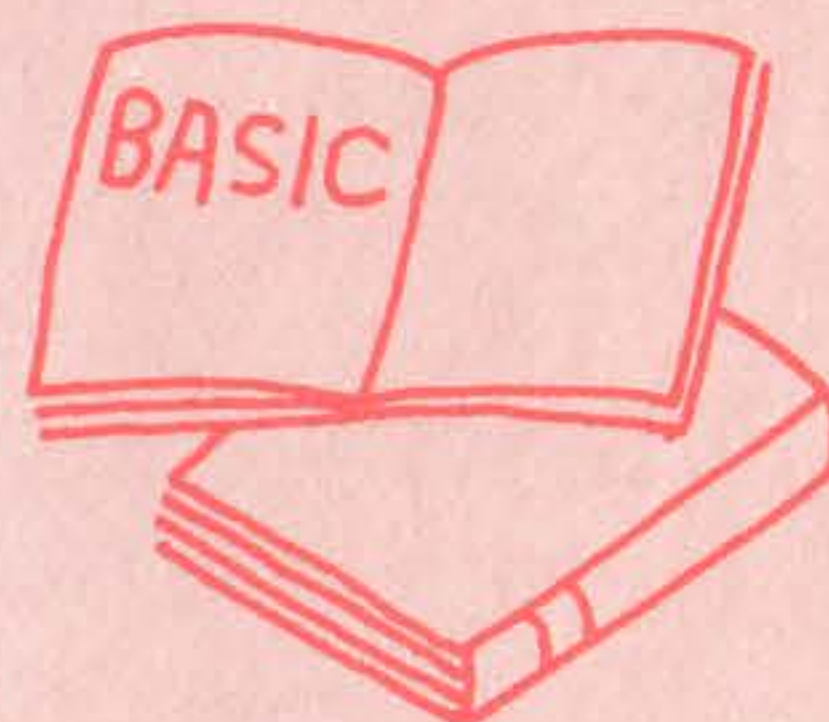
```




```

2170 PRESET (24,16)
2180 PRINT#1,"カ"メシ"ヨウニ BASICノコメント"カ"
2190 PRESET (32,28)
2200 PRINT#1,"テ"テキマスノテ" ソノコメント"ヲ"
2210 PRESET (32,40)
2220 PRINT#1,"キーボ"ート"テ" ニュウリョクシテワタ"サイ。"
2230 PRESET (24,52)
2240 PRINT#1,"カ"メシ"ヨウノキーボ"ート"ニハ アカイワツカ"
2250 PRESET (32,64)
2260 PRINT#1,"ヒョウシ"サレマス。
2270 PRESET (24,76)
2280 PRINT#1,"ハシ"メルトキハ スハ"ースキーヲ オシテワタ"サイ。"
2290 A$=INKEY$
2300 IF A$="" GOTO 2310 ELSE 2290
2310 LINE(12,8)-(248,90),4,BF
2320 PRESET (40,40)
2330 PRINT#1,"ワタ"イマ コメント"ヲヨミコンテ"イマスノテ"
2340 PRESET (40,60)
2350 PRINT#1,"シハ"ラク オマチワタ"サイ。"
2360 GOSUB3320
2370 A$=INKEY$:IF A$<>"" GOTO 2370
2380 LINE(12,8)-(248,90),4,BF
2390 LINE(80,28)-(180,40),15,B
2400 FOR I=1 TO KT
2410 LINE(80,58)-(180,70),2,BF
2420 VT$=CM$(CM(I))
2430 PRESET (100,30)
2440 PRINT#1,VT$
2450 SL=LEN(VT$)
2460 FOR J=1 TO SL
2470 MO$=MID$(VT$,J,1)
2480 IF MO$=" $" THEN GOSUB 3490:GOTO 2510
2490 FL=ASC(MO$)-97
2500 PUT SPRITE 2,(AD(FL,0),AD(FL,1)),8,2
2510 A$=INKEY$:IF A$="" GOTO2510
2520 PUT SPRITE 3,(0,209),0,0
2530 IF A$=MO$ GOTO 2560
2540 PLAY"C"
2550 GOTO 2510
2560 BEEP
2570 PRESET(92+J*8,60),2

```




```

2580 PRINT#1,MO$
2590 NEXT J
2600 COLOR 4
2610 PRESET (100,30)
2620 PRINT#1,VT$
2630 COLOR 15
2640 NEXT I
2650 LINE(12,8)-(248,77),4,BF
2660 PRESET (40,50)
2670 PRINT#1,"モウイチト" ヤリマスカ ? (Y/N)
2680 A$=INKEY$
2690 IF A$="y" GOTO 2150
2700 IF A$="n" THEN RETURN
2710 GOTO 2680
2720 'コメント"ニュウリョクノ テスト
2730 LINE(12,8)-(248,90),4,BF
2740 PRESET (24,16)
2750 PRINT#1,"コレハ カ"メンシ"ョウニ テ"テクル"
2760 PRESET (32,28)
2770 PRINT#1,"BASICノコメント"ヲ セ"ンフ"
2780 PRESET (32,40)
2790 PRINT#1,"ニュウリョクズルノニ ナンヒ"ョウ"
2800 PRESET (32,52)
2810 PRINT#1,"カカルカラ ハカルテストテ"ス。"
2820 PRESET (24,64)
2830 PRINT#1,"ハシ"メルトキハ スハ"ースキーヲ オシテクダ"サイ。"
2840 A$=INKEY$
2850 IF A$="" GOTO 2860 ELSE 2840
2860 LINE(12,8)-(248,90),4,BF
2870 PRESET (40,40)
2880 PRINT#1,"モンタ"イヲ サクセイチュウテ"ス。
2890 PRESET (40,60)
2900 PRINT#1,"シハ"ラク オマチクダ"サイ。"
2910 GOSUB3320
2920 A$=INKEY$:IF A$<>"" GOTO 2920
2930 LINE(12,8)-(248,90),4,BF
2940 TIME=0
2950 LINE(80,28)-(180,40),15,B
2960 FOR I=1 TO KT
2970 LINE(80,58)-(180,70),2,BF
2980 VT$=CM$(CM(I))

```




```

2990 PRESET (100,30)
3000 PRINT#1,VT$
3010 SL=LEN(VT$)
3020 FOR J=1 TO SL
3030 MO$=MID$(VT$,J,1)
3040 A$=INKEY$:IF A$="" GOTO 3040
3050 IF A$="$" THEN GOSUB 3490:GOTO 3090
3060 IF A$<"a" OR A$>"z" THEN 3040
3070 KI=ASC(MO$)-97
3080 PUT SPRITE 2,(AD(KI,0),AD(KI,1)),8,2
3090 IF A$=MO$ GOTO 3120
3100 PLAY"C"
3110 GOTO 3040
3120 BEEP
3130 PRESET (92+J*8,60),2
3140 NEXT J
3150 COLOR 4
3160 PRESET (100,30)
3170 PRINT#1,VT$
3180 COLOR 15
3190 PUT SPRITE 2,(0,209),0,0
3200 PUT SPRITE 3,(0,209),0,0
3210 NEXT I
3220 TI=TIME
3230 LINE(12,8)-(248,90),4,BF
3240 PRESET (24,28)
3250 PRINT#1,"タタ"イマノタイムハ";INT(TI/60);"ヒ"ョウデ"シタ"
3260 PRESET (40,60)
3270 PRINT#1,"モウイチ" ヤリマスカ ? (Y/N)
3280 A$=INKEY$
3290 IF A$="y" GOTO 2720
3300 IF A$="n" THEN RETURN
3310 GOTO 3280
3320 RESTORE 3650
3330 KT=0
3340 READ C$
3350 IF C$="END" GOTO 3390
3360 KT=KT+1
3370 CM$(KT)=C$
3380 GOTO 3340
3390 FOR I=1 TO KT

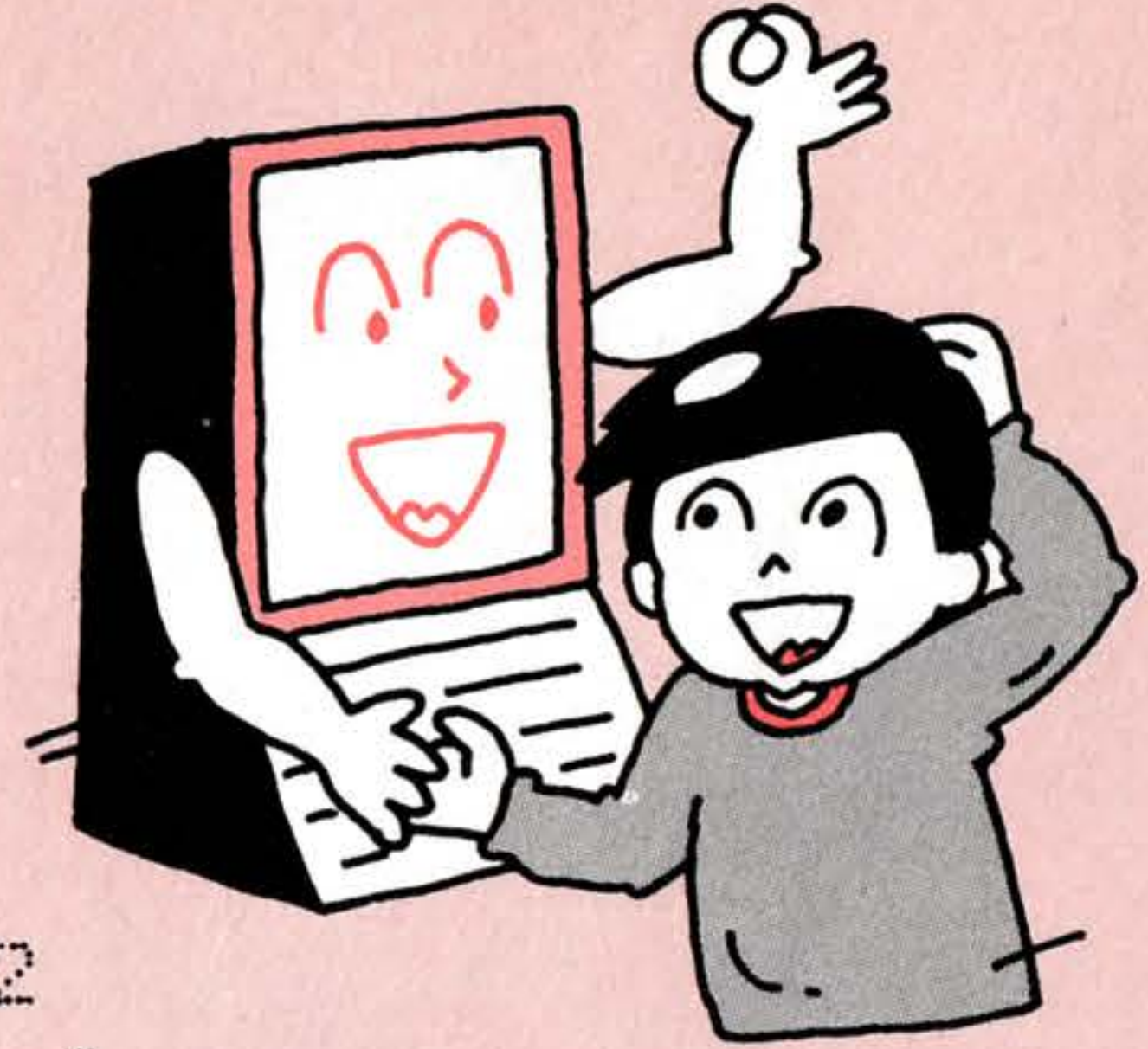
```




```

3400 CM(I)=0
3410 NEXT
3420 R=RND(-TIME)
3430 FOR I=1 TO KT
3440 RN=INT(RND(1)*KT+1)
3450 IF CM(RN)<>0 GOTO 3440
3460 CM(RN)=I
3470 NEXT
3480 RETURN
3490 PUT SPRITE 2,(72,99),8,2
3500 PUT SPRITE 3,(20,147),4,3
3510 RETURN
3520 'sprite data
3530 DATA &h18,&h3c,&h7e,&hff,&hff,&h7e,&h3c,&h18
3540 DATA &hfe,&h82,&h82,&h82,&h82,&h82,&hfe,0
3550 DATA &hff,&h80,&h80,&h80,&h80,&h80,&hff,0
3560 DATA 0,0,0,0,0,0,0,0
3570 DATA &hf8,&h8,&h8,&h8,&h8,&h8,&hf8,0
3580 DATA 40,131,112,147,80,147,72,131
3590 DATA 64,115,88,131,104,131,120,131
3600 DATA 144,115,136,131,152,131,168,131
3610 DATA 144,147,128,147,160,115,176,115
3620 DATA 32,115,80,115,56,131,96,115
3630 DATA 128,115,96,147,48,115,64,147
3640 DATA 112,115,48,147
3650 DATA asc,auto,beep,chr$,circle
3660 DATA clear,cload,close,cls,color
3670 DATA cont,csave,data,delete,dim
3680 DATA dim,draw,end,fix,for
3690 DATA next,gosub,goto,if,then
3700 DATA else,inkey$,input,int,left$
3710 DATA len,line,list,locate,lprint
3720 DATA mod,new,open,paint,play
3730 DATA point,preset,print,pset,put
3740 DATA read,rem,renum,restore,return
3750 DATA right$,mid$,rnd,run,screen
3760 DATA sound,space$,stick,sprite$,spc
3770 DATA stop,strig,string$,str$,tab
3780 DATA time,val,width,sprite,to
3790 DATA END

```



● さ く い ん

パソコンの用語 (アイウエオ順)

オフィスオートメーション (OA) ^{えいぎょう} 営業や^{じ む かん り} 事務管理、^{じょうほうしり} 情報処理やワー
ドプロセッサなどの^{じ む き き} 事務機器まで、^{しょくば} 職場の^{しごと} 仕事を^{ひと} 人にかわっ
てコンピュータに^{しり} 処理させる……………(11)

エラーメッセージ プログラム^{じょう} 上に^{ぶんぼうてき} 文法的な^{あやま} 誤りがあつた^{ば あい} 場合、その^{あやま} 誤
りのある^{ぎょうばんごう} 行番号とその^{しゅるい} 種類を^{が めん} 画面に^{ひょうじ} 表示して^し 知らせる…(83)

キーボード コンピュータにプログラムやデータなどを^{にゅうりよく} 入力する^{じゅうよう} 重要な
^{にゅうりよくそうち} 入力装置……………(17)(32)

行番号 ^{ぎょうばんごう} プログラムの^{じっこう} 実行や^{き おくじゅんじょ} 記憶順序を示すために、^{かくぎょう} 各行の^か 書き出しに
^{ばんごう} 付けられた番号……………(41)

サブルーチン プログラムの中で^{なか} 同じ^{おな} 仕事^{しごと} が^{なんかい} 何回も^で 出てくるような^{ば あい} 場合、
これをメインプログラムから^き 切りはなして^{べつ} 別に^{まとめた} まとめたプロ
グラム。その^{ぶぶん} 部分だけを、^よ 呼び出して^{なんかい} 何回でも^{つか} 使うことがで
きる……………(65)

16進数 ^{しんすう} 10進数は10を^{くぎ} 区切りとして^{けた} 桁が^あ 上がるが、^{しんすう} 16進数では10～15ま
でをA、B、C、D、E、Fを^{つか} 使って^{けた} 1桁であらわす…(21)

ジョイスティック ^{にゅうりよく き き} 入力機器のひとつで、ゲームの^{こま} 駒の^{い どう} 移動などに^{つか} 使う
バースイッチのこと……………(19)(27)

ダイレクトモード ^{めいれい} ひとつの^{き おく} 命令をメモリに^{じっこう} 記憶させずに^{じっこう} 実行すること。
^{めいれいぶん} 命令文には^{ぎょうばんごう} 行番号を^つ 付ける^{ひつよう} 必要はない……………(40)

ディスプレイ プログラムや^{しごと} 仕事の^{けつ か} 結果を^{ひょうじ} 表示する^{そうち} 装置……………(14)

2進数 ^{しんすう} オン／オフの^{でん き しんごう} 電気信号ですべてを^{しり} 処理するコンピュータでは、
0と1の^{すうじ} 数字であらゆる^{じょうほう} 情報を^{ひょうげん} 表現する……………(20)

バイト (byte) 8ビットを1バイトという……(22)

ビット (bit) 情報量の単位。2進数の桁数のこと。1ビットの情報は2進数1桁分だから、0か1である……(22)

ファンクションキー (function key) キーボード上で、プログラムの実行によく使われる命令(機能)を集めた専用キー……(34)

プログラム コンピュータにさせる仕事の手順を書いたもの…(18)(40)

プログラムモード いったんメモリへプログラムをぜんぶ記憶させてから実行する方法……(40)

フロッピーディスク プログラムを書き込んだり、また読み出してメモリに入力するための外部記憶装置……(19)(35)

変数 データを出し入れする特定の記憶場所。たくさんの変数を区別するために変数名を付ける。数値変数には数値データが、文字変数には、文字データが入る。文字変数には、変数名のうしろに\$ (ドルマーク) をつける……(51)

乱数 不規則な数。サイコロやルーレットもその一種。パソコンでは、RND関数で、乱数を発生させる……(98)

MSX・BASIC命令 (A B C 順)

AUTO (オート) 行番号や行番号の間隔を自動的に発生させる……(59)

CHR\$ (キャラクタダラー) キャラクタコードを指定して、文字や記号を求める……(159)

CIRCLE (サークル) 位置や色、大きさなどを指定して、画面に円やだ円、扇形などを描く……(125)

CLOAD (カセットロード) カセットテープに記録してあるプログラムを読み込んでコンピュータに入力する……(35)

- CLS(クリアスクリーン)** 画面^{がめん}に表示^{ひょうじ}されている文字^{もじ}や記号^{きごう}、グラフィックなどのすべてを消^けす……………(32)(40)
- COLOR (カラー)** 画面^{がめん}の表示色^{ひょうじしよく}、背景色^{はいけいしよく}、周辺色^{しゅうへんしよく}の色^{いろ}を指定^{してい}する……………(116)
- CSAVE (カセットセーブ)** プログラムをカセットに記録^{きろく}する……………(35)
- DELETE(デリート)** プログラムを行単位^{ぎょうたん い}で削除^{さくじょ}する……………(111)
- DRAW(ドロー)** 指定^{してい}した座標^{ざひょう}を出発点^{しゅつぱつてん}として、指定^{してい}した色^{いろ}で水平^{すいへい}、垂直^{すい}、斜め^{なな}の8方向^{ほうこう}に直線^{ちくせん}を引く。ひと筆描^{ふでが}きの要領^{ようりょう}で自由^{じゆう}な図形^{ずけい}を表現^{ひょうげん}することができる……………(140)
- FOR~NEXT(フォー~ネクスト)** FORとNEXTには含まれた命令^{めいれい}を指定^しの回数^{かいすう}だけくり返^{かえ}す……………(57)(68)
- GOTO(ゴーツー)** 強制的^{きやうせいてき}に指定^{してい}の行番号^{ぎやうばんごう}にジャンプさせる……………(62)
- GOSUB(ゴースブ)** 強制的^{きやうせいてき}に指定^{してい}の行番号^{ぎやうばんごう}にジャンプさせ、その行^{ぎやう}から命令^{めいれい}を実行^{じっこう}したあと、RETURN文^{りたーんぶん}でもとに戻^{もど}り、その次^{つぎ}の命令^{めいれい}から実行^{じっこう}する……………(64)
- IF~THEN~ELSE (イフ~ゼン~エルス)** 条件^{じょうけん}が成立^{せいりつ}しているかどうか調べ、条件^{じょうけん}しだいでプログラムを実行^{じっこう}する。IF~GOTOという形^{かたち}もある……………(72)
- INT (インテジャー)** 小数点^{しょうすうてん}の付^ついている数値^{すうち}の小数点^{しょうすうてん}以下^いを切り捨^きてて整数^{せいすうち}値にする。……………(98)
- INPUT(インプット)** キーボードからの入力^{にゅうりよく}を受け付^うける……………(88)
- KEY OFF(キーオフ)** ファンクションキーの内容^{ないよう}の表示^{ひょうじ}を画面^{がめん}から消^けす……………(107)
- LINE (ライン)** 2点間^{てんかん}に線^{せん}を引^ひいたり、2点間^{てんかん}の線^{せん}を対角線^{たいかくせん}とする四角形^{しかくけい}を描^かく……………(119)

- LIST (リスト)** プログラムを画面に表示させる……(44)
- LOCATE (ローケート)** X軸とY軸でカーソルの座標位置を指定する……
 ……………(48)
- NEW (ニュー)** 内部メモリに記憶してあるプログラムを消す……(59)
- OPEN (オープン)** ファイルにデータを書き出すときに、そのファイル
 を開く……(143)
- PAINT (ペイント)** 指定した境界色で囲んだ部分を指定した色で塗る……
 ……………(134)
- PLAY (プレイ)** 3重和音までの音楽を演奏する……(144)
- PSET (ポイントセット)** 指定した座標のドットを表示……(136)
- PRESET (ポイントリセット)** 指定した座標のドットを消す……(138)
- PRINT (プリント)** 計算結果や文字、記号などを画面に表示する……
 ……………(38)(42)
- PUT SPRITE (プットスプライト)** スプライトを表示させる……(162)
- READ~DATA (リード~データ)** DATA文にあるデータを呼び出して、
 READ文にある変数に代入する……(77)
- RENUM (リナンバー)** 行番号を整理して並べかえる……(111)
- RND (ランダム)** 数を不規則に発生させる……(98)
- RUN (ラン)** プログラムを実行させる……(40)
- SCREEN (スクリーン)** 使用する画面モードを選択する……(114)
- SOUND (サウンド)** レジスタにデータを与えて音を発生させる (153)
- SPRITE\$ (スプライトダラー)** スプライトパターンを定義する (160)
- WIDTH (ウイダス)** 画面に表示する桁数と行数を指定する……(60)

MSX

絵でわかるたのしいパソコン入門

1987年5月25日 発行 ©

著 者 関 口 泰 夫

発 行 者 富 永 弘 一

印 刷 所 公和印刷株式会社

発行所 東京都台東区 株式会社 新星出版社
台東2丁目24

電話 (831)0743 郵便番号110 振替東京 4-72233

ISBN4-405-06062-2

マイコン時代をリードする新星出版社の

わかりやすい即実戦・実用マイコンBooks

MSXシリーズ

定価各980円

MSX

☆山下 利秋著

はじめてのパソコン入門

はじめての人でも10日間のレッスンでマスターできるようまとめた

ゲームで覚える

☆川中 篤胤著

MSXベーシック

9種類のゲームを楽しみながら基本を理解できるようにまとめた

MSX

☆大沢昭二／田中一郎著

はじめてのプログラミング

基礎知識と短いプログラム例でコマンドとステートメントを紹介

よくわかる

☆山下 利秋著

ぼくらのパソコン入門

ゲームやプログラム例で小学生でも基礎知識が身につくように解説

☆田中一郎／小山郁夫著

MSXベーシック用語辞典

簡単なプログラム例と実行結果を中心にわかりやすく説明、解説

早わかり

関口 泰／山科敦之著

マイコン用語辞典

まんが

きぎょうへい／さとう光著

パソコンゼミナール

まんが

田中一郎／愛沢ひろし著

パソコンの一般知識

絵でわかる

新井克彦／こしあきお著

初歩のマイコン百科

Z80

若松登志樹著

わかる機械語入門

早わかり

春田正夫／沢田昭著

マシン語事典

三木 守著

初歩のパソコン入門

三宅誠／佐藤清明著

初歩のマイコン入門

新星出版社の
パソコン・マイコンシリーズ

まんが パソコンの一般知識
田中一郎
愛沢ひろし
パソコン内部のおおよその仕組み、情報の記憶のされ方、処理方法を説明した

まんが パソコンゼミナール
きぎようへい
さとう光
パソコンのだいたいの仕組み、操作法、ベーシックの基本をマンガで解説した

まんが ベーシックプログラミング
北園一平
愛沢ひろし
電話帳・住所録など、実務や家庭ですぐに役立つプログラムの作り方を解説した

早わかり ベーシック用語辞典
田中一郎
小山郁夫
ベーシックのコマンド、ステートメントの意味だけでなく、働き、使い方を説明

早わかり ベーシック決まり文句
大橋均
田中一郎
ステートメントの典型的な使い方・働きをサンプルプログラムと実行結果で説明

早わかり マイコン用語辞典
関口泰
山科敦之
単なるマイコン用語の解説にとどまらずマイコンについてのいろいろがわかる本

絵でわかる **MSX**
たのしいパソコン入門



新星出版社
定価980円



ISBN4-405-06062-2 C2055 ¥980E